

LINE#	LOC	CODE	LINE
-------	-----	------	------

00001	0000	; *****	
00002	0000	; *	
00003	0000	; * AAA DDDD DDDD 000 N N *	
00004	0000	; * A A D D D D 0 0 NN N *	
00005	0000	; * A A D D D D *** 0 0 NN N *	
00006	0000	; * AAAAA D D D D *** 0 0 NN N *	
00007	0000	; * A A D D D D 0 0 N NN *	
00008	0000	; * A A DDDD DDDD 000 N N *	
00009	0000	; *	
00010	0000	; * M M 000 N N *	
00011	0000	; * MM MM 0 0 NN N *	
00012	0000	; * *** M M 0 0 NN N *	
00013	0000	; * *** M M 0 0 NN N *	
00014	0000	; * M M 0 0 NN *	
00015	0000	; * M M 000 N N *	
00016	0000	; *	
00017	0000	; *****	

00019	0000	; *****	
00020	0000	; *	
00021	0000	; * 64K-ADD-ON MONITOR *	
00022	0000	; *	
00023	0000	; * THIS PROGRAM ALLOWS A USER TO *	
00024	0000	; * EXAMINE, MODIFY, AND RUN PROGRAMS *	
00025	0000	; * WITH BREAKPOINTS IN THE 64K ADD-ON *	
00026	0000	; * MEMORY. IT IS FUNCTIONALLY THE SAME *	
00027	0000	; * AS THE PET RESIDENT MONITOR. *	
00028	0000	; *	
00029	0000	; * 29JAN81 RJF 29JUL81 RJF *	
00030	0000	; *	
00031	0000	; * CBM PART # 8032118-01 REV B *	
00032	0000	; *	
00033	0000	; *****	

LINE#	LOC	CODE	LINE
00035	0000		;
00036	0000		; VIRTUAL REGISTERS
00037	0000		;
00038	0000	PCH ****+1	; PROGRAM COUNTER
00039	0001	PCL ****+1	
00040	0002	FLGS ****+1	; PROCESSOR STATUS
00041	0003	ACC ****+1	; ACCUMULATOR
00042	0004	XR ****+1	; X INDEX
00043	0005	YR ****+1	; Y INDEX
00044	0006	SP ****+1	; STACK POINTER
00045	0007	INVH ****+1	; USER IRQ VECTOR
00046	0008	INVL ****+1	
00047	0009	MEMMAP ****+1	; ADD-ON CONTROL REG
00048	000A		;
00049	000A		; INDIRECT POINTERS
00050	000A		;
00051	000A	STAL ****+1	; SAVE STORE POINTER
00052	000B	STAH ****+1	
00053	000C		;
00054	000C	SAL ****+1	; LOAD/SAVE START
00055	000D	SAH ****+1	
00056	000E		;
00057	000E	EAL ****+1	; LOAD/SAVE END
00058	000F	EAH ****+1	
00059	0010		;
00060	0010	TMPO ****+2	; MONITOR INDIRECTS
00061	0012		;
00062	0012	TMP2 ****+2	
00063	0014		;
00064	0014		; WORKING VARIABLES
00065	0014		;
00066	0014	TMPC ****+1	
00067	0015	SAVX ****+1	
00068	0016	WRAP ****+1	; WRAP FLAG FOR DISPLYM
00069	0017		;
00070	0017	TMPI ****+1	; I.A SAVE FOR IRQ
00071	0018	TMPPS ****+1	; P SAVE FOR IRQ

LINE#	LOC	CODE	LINE
00073	0019		;
00074	0019		CINV ==\$90
00075	0019		CBINV ==\$92
00076	0019		BUF ==\$200
00077	0019		FNADR ==\$DA
00078	0019		FNLEN ==\$D1
00079	0019		CR ==\$D
00080	0019		BAD ==\$100
00081	0019		STATUS ==\$96
00082	0019		FA ==\$D4
00083	0019		SA ==\$D3
00084	0019		RCLRCH ==\$F2A6
00085	0019		RLISTN ==\$F0D5
00086	0019		RSECND ==\$F143
00087	0019		RCIOUT ==\$F19E
00088	0019		RUNLSN ==\$F189
00089	0019		RTALK ==\$F0D2
00090	0019		RTKSA ==\$F193
00091	0019		RACPTR ==\$F1C0
00092	0019		RUNTLK ==\$F1AE
00093	0019		ROPENI ==\$F4A5
00094	0019		STKEY ==\$9B
00095	0019		NDX ==\$9E

LINE#	LOC	CODE	LINE
00097	0019		;
00098	0019		* =1024
00099	040C		SYS =158 ; BASIC TOKEN VALUE
00100	0400		;
00101	0400		; ENTER COMPILED BASIC TEXT
00102	0400		;
00103	0400	00	. BYT 0
00104	0401	0D 04	. WOR INIT-2
00105	0403	10 00	. WOR \$10
00106	0405	9E	. BYT SYS
00107	0406	2B 31	. BYT '(1039)', 0, 0, 0
00107	040C	00	
00107	040D	00	
00107	040E	00	
00108	040F		;
00109	040F		; INITIALIZE INTERRUPT PROCESS
00110	040F		;
00111	040F	78	INIT SEI
00112	0410	A9 80	LDA #X100000000
00113	0412	85 09	STA MEMMAP
00114	0414	8D F0 FF	STA \$FFFO
00115	0417	20 2B 04	JSR SETIRQ
00116	041A	A9 88	LDA #X10001000
00117	041C	8D F0 FF	STA \$FFFO
00118	041F	20 2B 04	JSR SETIRQ
00119	0422	A5 09	LDA MEMMAP
00120	0424	8D F0 FF	STA \$FFFO
00121	0427		;
00122	0427	5B	CLI
00123	0428	4C 7A 04	JMP TIMC

LINE#	LOC	CODE	LINE
00125	042B	A9 40	SETIRG LDA #CIRG
00126	042D	8D FE FF	STA \$FFFE
00127	0430	A9 04	LDA #>IRG
00128	0432	8D FF FF	STA \$FFFF
00129	0435	A9 59	LDA #<NMI
00130	0437	8D FA FF	STA \$FFFA
00131	043A	A9 04	LDA #>NMI
00132	043C	8D FB FF	STA \$FFFB
00133	043F	60	RTS
00135	0440		; PROCESS IRG
00136	0440		;
00137	0440	85 17	IRG STA TMPA ; PRESERVE . A
00138	0442		;
00139	0442	68	PLA
00140	0443	48	PHA
00141	0444	85 18	STA TMPPS
00142	0446		;
00143	0446	A9 00	LDA #X00000000
00144	0448	8D F0 FF	STA \$FFFO
00145	0448		;
00146	0448		; PUSH RETURN FROM INTERRUPT ADDRESS
00147	0448		;
00148	0448	A9 04	LDA #>RTIP
00149	044D	48	PHA
00150	044E	A9 72	LDA #<RTIP
00151	0450	48	PHA
00152	0451		;
00153	0451	A5 18	LDA TMPPS
00154	0453	48	PHA ; PUSH DUMMY STATUS
00155	0454		;
00156	0454	A5 17	LDA TMPA ; RESTORE . A
00157	0456		;
00158	0456		; GO TO ROM IRG SERVICE
00159	0456		;
00160	0456	6C FE FF	JMP (\$FFFE)
00162	0459		; PROCESS NMI
00163	0459		;
00164	0459		; PROCESS NMI
00165	0459		;
00166	0459	85 17	NMI STA TMPA ; PRESERVE . A
00167	045B		;
00168	045B	68	PLA
00169	045C	48	PHA
00170	045D	85 18	STA TMPPS
00171	045F		;

LINE#	LOC	CODE	LINE
00172	045F		;
00173	045F	A9 00	LDA #X00000000
00174	0461	6D F0 FF	STA \$FFFO
00175	0464		;
00176	0464		; PUSH RETURN FROM INTERRUPT ADDRESS
00177	0464		;
00178	0464	A9 04	LDA #>RTIF
00179	0466	48	PHA
00180	0467	A9 72	LDA #<RTIP
00181	0469	48	PHA
00182	046A		;
00183	046A	A5 18	LDA TMPPS
00184	046C	48	PHA ; PUSH DUMMY STATUS
00185	046D		;
00186	046D	A5 17	LDA TMPA
00187	046F		;
00188	046F		; GO TO ROM IRQ SERVICE
00189	046F		;
00190	046F	6C FA FF	JMP (\$FFFA)
00192	0472		; RETURN FROM INTERRUPT PROCESS
00193	0472		;
00194	0472	48	RTIP PHA
00195	0473		;
00196	0473		; MAP BACK TO ORIGINAL RAM
00197	0473		;
00198	0473	A5 09	LDA MEMMAP
00199	0475	8D F0 FF	STA \$FFFO
00200	0478		;
00201	0478		; RESTORE OLD .A
00202	0478		;
00203	0478	68	PLA
00204	0479		;
00205	0479		; BACK TO USER
00206	0479		;
00207	0479	40	RTI

LINE#	LOC	CODE	LINE
-------	-----	------	------

00209	047A		; *****
00210	047A		; *
00211	047A		; * KERNAL MONITOR
00212	047A		; *
00213	047A		; * ENTRY VIA CALL (JMP) OR BREAKPOINT (BRK)
00214	047A		; * ---FUNCTIONS---
00215	047A		; * <I> ALTER MEMORY
00216	047A		; * <I> ALTER REGISTERS
00217	047A		; * <R> DISPLAY REGISTERS
00218	047A		; * <M> DISPLAY MEMORY
00219	047A		; * <C> START EXECUTION OF CODE
00220	047A		; * <L> LOAD MEMORY
00221	047A		; * <S> SAVE MEMORY
00222	047A		; * <E> DISK COMMAND
00223	047A		; * <A> ADD-ON CONTROL REGISTER
00224	047A		; * <OTHER> LOAD AND EXECUTE FROM DISK
00225	047A		; *
00226	047A		; * FOR SYNTAX & SEMANTICS SEE CBM KERNAL MANUAL
00227	047A		; * COPYRIGHT (C) 1980 BY CBM
00228	047A		; *****

LINE#	LOC	CODE	LINE
00230	047A		; *****CALL ENTRY*****
00231	047A		;
00232	047A		TIMC
00233	047A	A9 88	LDA #<TIMB
00234	047C	85 92	STA CBINV
00235	047E	A9 04	LDA #>TIMB
00236	0480	85 93	STA CBINV+1
00237	0482	A9 0E	LDA #MS34-MS1 ; CALL ENTRY
00238	0484	85 14	STA TMPC
00239	0486	D0 10	BNE B3 ; BRANCH ALWAYS
00240	0488		;
00241	0488		; *****BREAK ENTRY*****
00242	0488		;
00243	0488	20 91 09	TIMB JSR CLRCH ; CLR CHANNELS
00244	0488	A9 75	LDA #MS36-MS1 ; BREAK ENTRY
00245	048D	85 14	STA TMPC
00246	048F	D8	CLD
00247	0490		;
00248	0490		; SAVE .Y, .X, .A, FLAGS, AND PC
00249	0490		;
00250	0490	A2 05	LDX #5
00251	0492	68	PLA
00252	0493	95 00	STA PCH, X
00253	0495	CA	DEX
00254	0496	10 FA	BPL B1
00255	0498		;
00256	0498	A5 90	B3 LDA CINV
00257	049A	85 08	STA INVL ; SAVE IRQ LOW
00258	049C	A5 91	LDA CINV+1
00259	049E	85 07	STA INVH ; SAVE IRQ HIGH
00260	04A0		;
00261	04A0	BA	TSX
00262	04A1	86 06	STX SP ; SAVE ORIGINAL SP
00263	04A3	58	CLI ; CLEAR INTS
00264	04A4		;
00265	04A4	A4 14	B5 LDY TMPC ; MESSAGE CODE
00266	04A6	20 62 09	JSR MSG ; PRINT BREAK/CALL
00267	04A9		;
00268	04A9	A9 52	LDA #'R ; DISPLAY REGS ON ENTRY
00269	04AB	D0 17	BNE S0 ; BRANCH ALWAYS
00270	04AD		;
00271	04AD		; *****ERROR ENTRY*****
00272	04AD		;
00273	04AD	20 50 05	ERROPR JSR OUTGST
00274	04B0	68	PLA
00275	04B1	68	PLA
00276	04B2		;
00277	04B2		; *****COMMAND INTERPRETER ENTRY*****
00278	04B2		;
00279	04B2		STRM1=-1
00280	04B2	A2 00	LDX #<BUF ; PLACE TO PUT FILE NAME
00281	04B4	A0 02	LDY #>BUF
00282	04B6	86 DA	STX FNADR
00283	04B8	84 DB	STY FNADR+1
00284	04BA	20 53 05	JSR CRLF

LINE#	LOC	CODE	LINE
00285	04BD		
00286	04BD	20 71 09	ST1 JSR BASIN ; READ COMMAND
00287	04C0	C9 20	CMF ##20
00288	04C2	F0 F9	BEG ST1 ; SPAN BLANKS
00289	04C4		
00290	04C4		; COMMAND INTERPRETER
00291	04C4		
00292	04C4	A2 00	S0 LDX #0
00293	04C6	86 16	STX WRAP
00294	04CB	86 D1	STX FNLEN
00295	04CA	AB	TAY ; SAVE CURRENT COMMAND
00296	04CB		
00297	04CB		; PUT RETURN ADDRESS FOR COMMANDS ON STACK
00298	04CB		
00299	04CB	A9 04	LDA #>STRTM1
00300	04CD	48	PHA
00301	04CE	A9 B1	LDA #<STRTM1
00302	04D0	48	PHA
00303	04D1		
00304	04D1	98	TYA ; CURRENT COMMAND IN .A
00305	04D2		
00306	04D2	DD 16 05	S1 CMP CMDS,X ; IS IT THIS ONE?
00307	04D3	DD 0F	BNE S2 ; NOT IT
00308	04D7		
00309	04D7	85 15	STA SAVX ; SAVE CURRENT COMMAND
00310	04D9		
00311	04D9		; INDIRECT JMP FROM TABLE
00312	04D9		
00313	04D9	BD 17 05	LDA CMDS+1,X
00314	04DC	85 10	STA TMP0
00315	04DE	BD 18 05	LDA CMDS+2,X
00316	04E1	85 11	STA TMP0+1
00317	04E3	6C 10 00	JMP (TMP0)
00318	04E6		
00319	04E6		; EACH TABLE ENTRY IS 3 LONG---SKIP TO NEXT
00320	04E6		
00321	04E6	E8	S2 INX
00322	04E7	E8	INX
00323	04E8	E8	INX
00324	04E9	EO 18	CPX #CMDEND-CMDS
00325	04EB	90 E5	BCC S1 ; LOOP FOR ALL COMMANDS
00326	04ED		
00327	04ED		; COMMAND NOT IN TABLE...LOOK ON DISK.
00328	04ED		; COMMAND NAME CAN BE ANY LENGTH AND
00329	04ED		; HAVE PARAMETERS.
00330	04ED		
00331	04ED	A2 00	LDX #0 ; LENGTH TO ZERO
00332	04EF	C9 0D	CMF ##D ; END OF NAME?
00333	04F1	F0 0D	BEG S4 ; YES...
00334	04F3	C9 20	CMF ##20 ; BLANK?
00335	04F5	F0 09	BEG S4 ; YES
00336	04F7	9D 00 02	STA BUF,X
00337	04FA	20 71 09	JSR BASIN ; GET NEXT
00338	04FD	E8	INX ; COUNT CHAR
00339	04FE	DD EF	BNE S3 ; AND CONTINUE

LINE# LOC CODE LINE

```

00340 0500      ;
00341 0500 85 14   S4   STA TMPC
00342 0502 8A      TXA      ;COUNT
00343 0503 F0 10   BEQ S6   ;IS ZERO
00344 0505      ;
00345 0505 85 D1   STA FNLEN
00346 0507 A9 08   LDA #8
00347 0509 85 D4   STA FA      ;WILL USE DEVICE #8
00348 050B 20 96 07 JSR LOAD  ;TRY TO LOAD COMMAND
00349 050E B0 05   BCS S6   ;BAD LOAD...
00350 0510      ;
00351 0510 A5 14   LDA TMPC  ;PASS LAST CHARACTER
00352 0512 6C 0A 00 JMP (STAL) ;GO DO IT
00353 0515      ;
00354 0515 60      S6   RTS

```

```

00356 0516 3A      CMDS  .BYT ':'  ;ALTER MEMORY
00357 0517 0F 06   .WOR ALTM
00358 0519 3B      .BYT ':'  ;ALTER REGISTERS
00359 051A F5 05   .WOR ALTR
00360 051C 52      .BYT 'R'  ;DISPLAY REGISTERS
00361 051D 73 05   .WOR DSPLYR
00362 051F 4D      .BYT 'M'  ;DISPLAY MEMORY
00363 0520 B4 05   .WOR DSPLYM
00364 0522 47      .BYT 'G'  ;START EXECUTION
00365 0523 32 06   .WOR GO
00366 0525 4C      .BYT 'L'  ;LOAD MEMORY
00367 0526 5E 06   .WOR LD
00368 0528 53      .BYT 'S'  ;SAVE MEMORY
00369 0529 5E 06   .WOR LD
00370 052B 2A      .BYT '*'  ;MAP MEMORY
00371 052C 2C 06   .WOR MAPPER
00372 052E 40      .BYT '@'  ;DISK COMMAND (ALTERNATE)
00373 052F 48 07   .WOR DISK
00374 0531      CMDEND

```

```

00376 0531 A5 10   PUTP  LDA TMPO  ;MOVE TMPO TO PCH,PCL
00377 0533 85 01   STA PCL
00378 0535 A5 11   LDA TMPO+1
00379 0537 85 00   STA PCH
00380 0539 60      RTS

```

```

00382 053A A9 02   SETR  LDA #<FL05 ;SET TO ACCFS REQ5
00383 053C 85 10   STA TMPO
00384 053E A9 00   LDA #>FL05

```

LINE#	LOC	CODE	LINE
00385	0540	85 11	STA TMP0+1
00386	0542	A9 05	LDA #5
00387	0544	60	RTS
00389	0545		; PRINTS ':' OR ';' BEFORE DATA TO PERMIT
00390	0545		; ALTER AFTER 'M' OR 'R' COMMAND
00391	0545		;
00392	0545	48	ALTRIT PHA ; PRESERVE ALTER CHARACTER
00393	0546	20 53 05	JSR CRLF
00394	0549	68	PLA
00395	054A	20 B1 09	JSR BSOUT
00397	054D	A9 20	SPACE LDA #\$20 ; OUTPUT A SPACE
00398	054F	2C	.BYT \$2C ; SKIP TWO BYTES
00399	0550	A9 3F	OUTGST LDA #'?' ; OUTPUT QUESTION
00400	0552	2C	.BYT \$2C ; SKIP TWO BYTES
00401	0553	A9 0D	CRLF LDA #\$D ; DO CARRIAGE RETURN
00402	0555	4C B1 09	JMP BSOUT
00404	0558		; DATA FOR REGISTER DISPLAY HEADING
00405	0558		;
00406	0558	0D	REGK .BYT CR, \$20, \$20 ; 3 SPACES
00406	0559	20	
00406	055A	20	
00407	055B	20 50	.BYT ' PC ', ' IRQ ', ' SR AC XR YR SP'
00407	055F	20 49	
00407	0564	20 53	
00409	0573		; DISPLAY REGISTER FUNCTION
00410	0573		;
00411	0573	A2 00	DSPLYR LDX #0
00412	0575	8D 58 05	D2 LDA REGK, X
00413	0578	20 B1 09	JSR BSOUT ; PRINT HEADING
00414	0578	E8	INX
00415	057C	E0 18	CPX #DSPLYR-REGK, MAX LENGTH
00416	057E	D0 F5	BNE D2
00417	0580	A9 38	LDA #';
00418	0582	20 45 05	JSR ALTRIT ; ALLOW ALTER AFTER DISPLAY
00419	0585	A6 00	LDX PCH
00420	0587	A4 01	LDY PCL
00421	0589	20 DC 06	JSR WRDA ; PRINT PROGRAM COUNTER

LINE#	LOC	CODE	LINE
00422	058C	20 4D 05	JSR SPACE
00423	058F	A6 07	LDX INVH
00424	0591	A4 08	LDY INVL
00425	0593	20 DC 06	JSR WROA ;PRINT IRQ VECTOR
00426	0596	20 3A 05	JSR SETR ;SET TO PRINT .P..A..X..Y..S
00427	0599		;
00428	0599		; DISPLAY MEMORY SUBROUTINE
00429	0599		;
00430	0599	85 14	DM STA TMPC ;BYTE COUNT
00431	059B	A0 00	LDY #0 ;INDIRECT INDEX
00432	059D	20 4D 05	DM1 JSR SPACE ;SPACE TWEEN BYTES
00433	05A0	B1 10	LDA (TMPC)Y
00434	05A2	20 E1 06	JSR WROB ;WRITE BYTE OF MEMORY
00435	05A5		;
00436	05A5		; INCREMENT INDIRECT
00437	05A5		;
00438	05A5	E6 10	INC TMPO
00439	05A7	D0 06	BNE DM2
00440	05A9	E6 11	INC TMPO+1
00441	05AB	D0 02	BNE DM2
00442	05AD	E6 16	INC WRAP
00443	05AF		;
00444	05AF	C6 14	DM2 DEC TMPC ;COUNT BYTES
00445	05B1	D0 EA	BNE DM1 ;UNTIL ZERO
00446	05B3	60	RTS
00448	05B4		; DISPLAY MEMORY FUNCTION
00449	05B4		;
00450	05B4	20 09 07	DSPLYM JSR RDOA ;READ START ADR
00451	05B7	80 39	BCS ERRS1 ;ERR IF NO SA
00452	05B9	20 F9 06	JSR T2T2 ;SA TO TMP2
00453	05BC		;
00454	05BC		; ALLOW USER TO TYPE JUST ONE ADDRESS
00455	05BC		;
00456	05BC	20 09 07	JSR RDOA ;READ END ADR
00457	05BF	90 06	BCC DSP10 ;GOOD.. NO DEFAULT
00458	05C1		;
00459	05C1	A5 12	LDA TMP2
00460	05C3	85 10	STA TMPO ;DEFAULT LOW BYTE
00461	05C5	A5 13	LDA TMP2+1
00462	05C7	85 11	STA TMPO+1 ;DEFAULT HI BYTE
00463	05C9		;
00464	05C9	20 F9 06	DSP10 JSR T2T2 ;SA TO TMPO, EA TO TMP2
00465	05CC	20 AA 08	JSR STOP ;STOP KEY?
00466	05CF	F0 20	BEG BEGS1 ;YES...BREAK LIST
00467	05D1		;
00468	05D1	A9 3A	LDA #';
00469	05D3	20 45 05	JSR ALTRIT ;ALLOW ALTER
00470	05D6	A6 11	LDX TMPO+1
00471	05D8	A4 10	LDY TMPO
00472	05DA	20 DC 06	JSR WROA ;WRITE START ADDRESS

LINE#	LOC	CODE	LINE
00473	05DD	A9 08	LDA #8 ; COUNT OF BYTES
00474	05DF	20 99 05	JSR DM ; DISPLAY BYTES
00475	05E2	A5 16	LDA WRAP
00476	05E4	D0 08	BNE BEGS1
00477	05E6		;
00478	05E6		; CHECK FOR END OF DISPLAY
00479	05E6		;
00480	05E6	38	SEC
00481	05E7	A9 12	LDA TMP2
00482	05E9	E5 10	SBC TMP0
00483	05EB	A5 13	LDA TMP2+1
00484	05ED	E5 11	SBC TMP0+1
00485	05EF	B0 DB	BCS DSP1 ; END >= START
00486	05F1		;
00487	05F1	40	BEQS1 RTS ; A. O. K. EXIT
00488	05F2		;
00489	05F2	4C AD 04	ERRS1 JMP ERROPR ; SYNTAX ERROR
00491	05F5		; ALTER REGISTER FUNCTION
00492	05F5		;
00493	05F5	20 09 07	ALTR JSR RDOA ; READ NEW PC
00494	05F8	B0 F8	BCS ERRS1 ; ERROR... NO ADDRESS
00495	05FA		;
00496	05FA	20 31 05	JSR PUTP ; ALTER PC
00497	05FD		;
00498	05FD	20 09 07	JSR RDOA ; READ NEW IRQ
00499	0600	B0 F0	BCS ERRS1 ; ERROR... NO ADDRESS
00500	0602		;
00501	0602	A5 10	LDA TMP0
00502	0604	85 08	STA INVL ; ALTER IRQ ECTOR
00503	0606	A5 11	LDA TMP0+1
00504	0608	85 07	STA INVH
00505	060A		;
00506	060A	20 3A 05	JSR SETR ; SET TO ALTER R'S
00507	060D	D0 07	BNE A4 ; BRANCH ALWAYS
00509	060F		; ALTER MEMORY - READ ADR AND DATA
00510	060F		;
00511	060F	20 09 07	ALTM JSR RDOA ; READ ALTER ADR
00512	0612	B0 DE	BCS ERRS1 ; IF SPACE, ERR
00513	0614		;
00514	0614	A9 08	LDA #8 ; ALLOW 8 BYTES CHANGE
00515	0616		;
00516	0616		; COMMON CODE FOR '1' AND '2'
00517	0616		;
00518	0616	85 14	A4 STA TMPC ; NUMBER OF BYTES TO CHANGE
00519	0618		;

LINE# LOC CODE LINE

```

00520 0618 20 16 07 A5 JSR RDOB ; READ BYTE
00521 0618 B0 0E BCS A9 ; NONE... END OF LINE
00522 061D ;
00523 061D A2 00 LDX #0
00524 061F 81 10 STA (TMPO,X) ; STORE IT AWAY
00525 0621 ;
00526 0621 ; INCREMENT STORE ADDRESS
00527 0621 ;
00528 0621 E6 10 INC TMPO
00529 0623 D0 02 BNE A6
00530 0625 E6 11 INC TMPO+1
00531 0627 ;
00532 0627 C6 14 A6 DEC TMPC ; COUNT BYTE
00533 0629 D0 ED BNE A5 ; UNTIL ZERO
00534 062B ;
00535 062B 60 A9 RTS

```

```

00537 062C ; MAP MEMORY
00538 062C ;
00539 062C 20 16 07 MAPPER JSR RDOB
00540 062F B5 09 STA MEMMAP
00541 0631 60 RTS

```

```

00543 0632 ; START EXECUTION FUNCTION
00544 0632 ;
00545 0632 20 45 07 G0 JSR RDOC ; SEE IF DEFAULT
00546 0635 F0 08 BEQ Q1 ; YES... PC IS ADDRESS
00547 0637 ;
00548 0637 20 09 07 JSR RDOA ; NO... GET NEW ADDR
00549 063A B0 1F BCS ERR1 ; ERROR... ADDRESS SCREWED UP
00550 063C ;
00551 063C 20 31 05 JSR PUTP ; MOVE ADDR TO P.C.
00552 063F ;
00553 063F A6 06 G1 LDX SP ; ORIG OR NEW SP VALUE TO SP
00554 0641 9A TXS
00555 0642 ;
00556 0642 78 SEI ; PREVENT DISASTER
00557 0643 ;
00558 0643 A5 07 LDA INVH
00559 0645 B5 91 STA CINV+1 ; SET UP IRQ VECTOR
00560 0647 A5 08 LDA INVL
00561 0649 B5 90 STA CINV
00562 064B ;
00563 064B ; GET FLAGS, PCH, PCL... A, X, Y
00564 064B ;
00565 064B A2 00 LDX #0
00566 064D B5 00 G2 LDA PCH, X

```

LINE#	LOC	CODE	LINE
-------	-----	------	------

00567	064F	48	PHA	; EVERYBODY ON STACK
00568	0650	EB	INX	
00569	0651	E0 06	CPX #6	
00570	0653	D0 FB	BNE Q2	
00571	0655			
00572	0655		; INTERRUPT RETURN SETS EVERYBODY UP	
00573	0655		; FROM DATA ON STACK	
00574	0655			
00575	0655	68	PREND PLA	
00576	0656	A8	TAY	
00577	0657	68	PLA	
00578	0658	AA	TAX	
00579	0659	68	PLA	
00580	065A	40	RTI	

00582	065B	4C AD 04	ERRL JMP ERRORPR	; SYNTAX ERROR JUMP
-------	------	----------	------------------	---------------------

00584	065E		; LOAD RAM FUNCTION	
00585	065E			
00586	065E	A0 01	LD LDY #1	
00587	0660	B4 D4	STY FA	; DEFAULT DEVICE #1
00588	0662	88	DEY	; Y=0 TO COUNT NAME LENGTH
00589	0663			
00590	0663	20 45 07	L1 JSR RDOC	; DEFAULT?
00591	0666	F0 1C	BEG L5	; YES... TRY LOAD
00592	0668			
00593	0668	C9 20	CMP #'	
00594	066A	F0 F7	BEG L1	; SPAN BLANKS
00595	066C			
00596	066C	C9 22	CMP #' "	; STRING NEXT?
00597	066E	D0 EB	L2 BNE ERRL	; NO FILE NAME ..
00598	0670			
00599	0670	20 45 07	L3 JSR RDOC	; GET CHARACTER OF NAME
00600	0673	F0 0F	BEG L5	; END... ASSUME LOAD
00601	0675			
00602	0675	C9 22	CMP #' "	; END OF STRING?
00603	0677	F0 14	BEG L6	; YES... COULD STILL BE 'L' OR
00604	0679			
00605	0679	91 DA	STA (FNADR)Y	; STORE NAME
00606	067B	E6 D1	INC FNLEN	
00607	067D	CB	INY	
00608	067E	C0 10	CPY #16	; MAX FILE NAME LENGTH
00609	0680			
00610	0680	FC D9	L4 BEG ERRL	; FILE NAME TOO LONG
00611	0682	D0 EC	BNE L3	; BRANCH ALWAYS
00612	0684			
00613	0684		; SEE IF WE GOT A LOAD	
00614	0684			
00615	0684	A5 15	L5 LDA SAVX	; GET LAST COMMAND
00616	0686	C9 4C	CMP #'L	
00617	0688	D0 E4	BNE L2	; NO. NOT A LOAD. ERROR

MONITOR PAGE 0016

LINE#	LOC	CODE	LINE		
00618	068A				
00619	068A	4C 96 07		JMP LOAD	; YES... DO LOAD
00620	068D				
00621	068D	20 45 07	L8	JSR RDOC	; MORE STUFF?
00622	0690	F0 F2		BEG L5	; NO... DEFAULT LOAD
00623	0692				
00624	0692	C9 2C		CMP #'	; DELEIMETER?
00625	0694	D0 D8	L9	BNE L2	; NO... BAD SYNTAX
00626	0696				
00627	0696	20 16 07		JSR RDOB	; YES... GET NEXT PARM
00628	0699	B0 3E		BCS L15	; NOT GOOD
00629	069B				
00630	069B	B5 D4		STA FA	
00631	069D				
00632	069D	20 45 07		JSR RDOC	; MORE PARMS?
00633	06A0	F0 E2		BEG L5	; NO... DEFAULT LOAD
00634	06A2				
00635	06A2	C9 2C		CMP #'	; DELIMETER?
00636	06A4	D0 EE	L12	BNE L9	; NO... BAD SYNTAX
00637	06A6				
00638	06A6	20 09 07		JSR RDOA	; START ADDRESS?
00639	06A9	B0 2E		BCS L15	; NO... BAD
00640	06AB				
00641	06AB	20 F9 06		JSR T2T2	; PRESERVE START
00642	06AE	20 71 09		JSR BASIN	; DELIMETER?
00643	06B1	C9 2C		CMP #'	
00644	06B3	D0 EF	L13	BNE L12	; NO...
00645	06B5	20 09 07		JSR RDOA	; TRY TO READ END
00646	06B8	B0 1F		BCS L15	; NONE... ERROR
00647	06BA				
00648	06BA				
00649	06BA				
00650	06BA	A5 10		LDA TMPO	
00651	06BC	B5 0E		STA EAL	
00652	06BE	A5 11		LDA TMPO+1	
00653	06C0	B5 0F		STA EAH	
00654	06C2	20 F9 06		JSR T2T2	
00655	06C5				
00656	06C5	20 71 09	L20	JSR BASIN	
00657	06C8	C9 20		CMP ##20	
00658	06CA	F0 F9		BEG L20	; SPAN BLANKS
00659	06CC				
00660	06CC	C9 0D		CMP #CR	
00661	06CE	D0 E3	L14	BNE L13	; MISSING CR AT END
00662	06D0	A5 15		LDA SAVX	; WAS COMMAND SAVE?
00663	06D2	C9 53		CMP #'S	
00664	06D4	D0 F8		BNE L14	; NO... LOAD CAN'T HAVE PARMS
00665	06D6	4C 2D 08		JMP SAVE	
00666	06D9				
00667	06D9	4C AD 04	L15	JMP ERROPR	

LINE#	LOC	CODE	LINE
00669	06DC		; WRITE ADR FROM TMP0 STORES
00670	06DC		;
00671	06DC	8A	WROA TXA ; HI-BYTE
00672	06DD	20 E1 06	JSR WROB
00673	06E0	9B	TYA ; LOW-BYTE
00675	06E1		; WRITE BYTE --- A = BYTE
00676	06E1		; UNPACK BYTE DATA INTO TWO ASCII
00677	06E1		; CHARACTERS. A=BYTE; X, A=CHARS
00678	06E1	4B	WROB PHA
00679	06E2	4A	LSR A
00680	06E3	4A	LSR A
00681	06E4	4A	LSR A
00682	06E5	4A	LSR A
00683	06E6	20 ED 06	JSR ASCII ; CONVERT TO ASCII
00684	06E9	AA	TAX
00685	06EA	6B	PLA
00686	06EB	29 0F	AND #\$0F
00688	06ED		; CONVERT NYBBLE IN A TO ASCII AND
00689	06ED		; PRINT IT
00690	06ED		;
00691	06ED	1B	ASCII CLC
00692	06EE	69 F6	ADC #\$F6
00693	06F0	90 02	BCC ASC1
00694	06F2	69 06	ADC #\$06
00695	06F4	69 3A	ASC1 ADC #\$3A
00696	06F6	4C B1 09	JMP BGOUT
00698	06F9		; EXCHANGE TEMPORARIES
00699	06F9		;
00700	06F9	A2 02	T2T2 LDX #2
00701	06FB	B5 0F	T2T21 LDA TMP0-1, X
00702	06FD	4B	PHA
00703	06FE	B5 11	LDA TMP2-1, X
00704	0700	95 0F	STA TMP0-1, X
00705	0702	6B	PLA
00706	0703	95 11	STA TMP2-1, X
00707	0705	CA	DEX
00708	0706	D0 F3	BNE T2T21
00709	070B	60	RTS

00711	0709		; READ HEX ADR. RETURN HI IN TMP0.
-------	------	--	------------------------------------

LINE#	LOC	CODE	LINE
00712	0709		; LO IN TMP0+1, AND CY=1
00713	0709		; IF SP CY=0
00714	0709		;
00715	0709	20 16 07	RDOA JSR RDOB ; READ 2-CHAR BYTE
00716	070C	B0 07	BCS RDOA2 ; SPACE
00717	070E	B5 11	STA TMP0+1
00718	0710	20 16 07	JSR RDOB
00719	0713	B5 10	STA TMP0
00720	0715	60	RDOA2 RTS
00721	0716		; READ HEX BYTE AND RETURN IN A
00722	0716		; AND CY=0 IF SP CY=1
00723	0716	A9 00	RDOB LDA #0 ; SPACE
00724	0718	BD 00 01	STA BAD ; READ NEXT CHAR
00725	0718	20 45 07	JSR RDOC
00726	071E	F0 19	BEG RDOB4 ; FAIL ON CR
00727	0720	C9 20	CMF #' ; BLANK?
00728	0722	F0 F2	BEG RDOB ; SPAN BLANKS...
00729	0724		;
00730	0724	20 3A 07	JSR HEXIT ; CONVERT TO HEX NYBBLE
00731	0727	0A	ASL A
00732	0728	0A	ASL A
00733	0729	0A	ASL A
00734	072A	0A	ASL A
00735	072B	BD 00 01	STA BAD
00736	072E	20 45 07	JSR RDOC ; 2ND CHAR ASSUMED HEX
00737	0731	F0 06	BEG RDOB4 ; FAIL ON CR
00738	0733	20 3A 07	JSR HEXIT
00739	0736	0D 00 01	DRA BAD
00740	0739		;
00741	0739	60	RDOB4 RTS
00743	073A		; CONVERT ASCII CHAR TO HEX NYBBLE
00744	073A		;
00745	073A	C9 3A	HEXIT CMP ##3A
00746	073C	08	PHP ; SAVE FLAGS
00747	073D	29 0F	AND ##0F
00748	073F	28	PLP
00749	0740	90 02	BCC HEX09 ; 0-9
00750	0742	69 08	ADC #8 ; ALPHA ADD B+CY=9
00751	0744	60	HEX09 RTS
00753	0745		; GET CHARACTER AND TEST FOR CR
00754	0745		;
00755	0745	20 71 09	RDOC JSR BASIN
00756	0748	C9 0D	CMF ##0D ; IS IT A CR
00757	074A	60	RTS ; RETURN WITH FLAGS

LINE# LOC CODE LINE

```

00759 074B      ; SEND DISK COMMAND OR READ STATUS
00760 074B      ;
00761 074B A9 00 DISK LDA #0      ; CLEAR STATUS @ I/O BEGIN
00762 074D 85 96 STA STATUS
00763 074F      ;
00764 074F 20 45 07 JSR RDOC      ; SEE IF STATUS CHECK
00765 0752 F0 22 BEQ DISK20    ; YES
00766 0754      ;
00767 0754 48      PHA
00768 0755 A9 08 LDA #8      ; FLOPPY IS DEVICE #8
00769 0757 20 9F 09 JSR LISTN    ; TELL FLOPPY TO RECEIVE
00770 075A A9 6F LDA #15+*60
00771 075C 20 AF 09 JSR SECND    ; ON COMMAND CHANNEL
00772 075F      ;
00773 075F 68      PLA
00774 0760 A6 96 LDX STATUS    ; ERROR?
00775 0762 10 06 BPL DISK15    ; NO...OK
00776 0764      ;
00777 0764 4C C6 0B DISK5 JMP ERROR5 ; DEVICE NOT PRESENT
00778 0767      ;
00779 0767 20 71 09 DISK10 JSR BASIN ; GET A CHARACTER
00780 076A C9 0D DISK15 CMP #*D ; SEE IF END
00781 076C 08      PHP        ; SAVE FOR LATER
00782 076D 20 BF 09 JSR CIOUT   ; OUT TO FLOPPY
00783 0770 28      PLP        ; END?
00784 0771 D0 F4 BNE DISK10    ; NO...CONTINUE
00785 0773      ;
00786 0773 4C CF 09 JMP UNLSN   ; YES...FLOPPY DONE
00787 0776      ;
00788 0776 20 53 05 DISK20 JSR CRLF ; FLOPPY IS DEVICE #8
00789 0779 A9 08 LDA #8      ; TELL FLOPPY TO SPEAK
00790 077B 20 DD 09 JSR TALK    ;
00791 077E A9 6F LDA #15+*60
00792 0780 20 ED 09 JSR TKSA    ; FROM ERROR CHANNEL
00793 0783      ;
00794 0783 A5 96 LDA STATUS    ; AN ERROR?
00795 0785 30 DD BMI DISK5     ; YES...
00796 0787      ;
00797 0787 20 FD 09 DISK25 JSR ACPTR ; GET A CHARACTER
00798 078A C9 0D CMP #*D      ; SEE IF END
00799 078C 08      PHP        ; TEST LATER
00800 078D 20 81 09 JSR BSOUT   ; OUT TO SCREEN
00801 0790 28      PLP        ; END?
00802 0791 D0 F4 BNE DISK25    ; NO...
00803 0793 4C 0D 0A JMP UNLTK ; YES... FLOPPY DONE

```

LINE#	LOC	CODE	LINE
00805	0796		; *****
00806	0796		; * LOAD RAM FUNCTION *
00807	0796		; *
00808	0796		; * LOADS FROM CASSETTE 1 OR 2, OR *
00809	0796		; * IEEE BUS DEVICES >=4 TO 31 AS *
00810	0796		; * DETERMINED BY CONTENTS OF *
00811	0796		; * VARIABLE FA. VERIFY FLAG IN .A *
00812	0796		; * HIGH LOAD RETURN IN X,Y. *
00813	0796		; * .A=0 PERFORMS LOAD,<> IS VERIFY*
00814	0796		; *
00815	0796		; *****
00817	0796		LOAD
00818	0796	A9 00	LDA #0
00819	079B	B5 96	STA STATUS
00820	079A		;
00821	079A	A5 D4	LDA FA ;CHECK DEVICE NUMBER
00822	079C	C9 04	CMP #4
00823	079E	B0 03	BCS LD20
00824	07A0		;
00825	07A0	4C D2 0B	LD10 JMP ERROR9 ;BAD DEVICE #
00826	07A3		;
00827	07A3		LD20
00828	07A3		;
00829	07A3		;LOAD FROM CBM IEEE DEVICE
00830	07A3		;
00831	07A3	A9 60	LDA #\$60 ;SPECIAL LOAD COMMAND
00832	07A5	B5 D3	STA SA
00833	07A7		;
00834	07A7	A4 D1	LDY FNLEN ;MUST HAVE FILE NAME
00835	07A9	D0 03	BNE LD25 ;YES...OK
00836	07AB		;
00837	07AB	4C CF 0B	JMP ERRORB ;MISSING FILE NAME
00838	07AE		;
00839	07AE	20 0D 0B	LD25 JSR LUKING ;TELL USER LOOKING
00840	07B1	20 1B 0A	JSR OPENI ;OPEN THE FILE
00841	07B4		;
00842	07B4	A5 D4	LDA FA
00843	07B6	20 DD 09	JSR TALK ;ESTABLISH THE CHANNEL
00844	07B9	A5 D3	LDA SA
00845	07BB	20 ED 09	JSR TKSA ;TELL IT TO LOAD
00846	07BE		;
00847	07BE	20 FD 09	JSR ACPTR ;GET FIRST BYTE
00848	07C1	B5 0E	STA EAL
00849	07C3	B5 0A	STA STAL
00850	07C5		;
00851	07C5	A5 96	LDA STATUS ;TEST STATUS FOR ERROR
00852	07C7	4A	LSR A
00853	07C8	4A	LSR A
00854	07C9	B0 39	BCS LD90 ;FILE NOT FOUND...
00855	07CB	20 FD 09	JSR ACPTR
00856	07CE	B5 0F	STA EAH
00857	07D0	B5 0B	STA STAH

```

LINE# LOC CODE LINE
00858 07D2 ;
00859 07D2 20 28 08 JSR LODING ; TELL USER LOADING
00860 07D5 ;
00861 07D5 A9 FD LD40 LDA ##FD ; MASK OFF TIMEOUT
00862 07D7 25 96 AND STATUS
00863 07D9 85 96 STA STATUS
00864 07DB ;
00865 07DB 20 AA 08 JSR STOP ; STOP KEY?
00866 07DE D0 03 BNE LD45 ; NO...
00867 07E0 ;
00868 07E0 4C 78 08 JMP BREAK ; STOP KEY PRESSED
00869 07E3 ;
00870 07E3 20 FD 09 LD45 JSR ACPTR ; GET BYTE OFF IEEE
00871 07E6 AA TAX
00872 07E7 A5 96 LDA STATUS ; WAS THERE A TIMEOUT?
00873 07E9 4A LSR A
00874 07EA 4A LSR A
00875 07EB B0 EB BCS LD40 ; YES... TRY AGAIN
00876 07ED 8A TXA
00877 07EE A0 00 LDY #0
00878 07F0 91 0E STA (EAL)Y
00879 07F2 E6 0E LD60 INC EAL ; INCREMENT STORE ADDR
00880 07F4 D0 02 BNE LD64
00881 07F6 E6 0F INC EAH
00882 07F8 24 96 LD64 BIT STATUS ; EQI?
00883 07FA 50 D9 BVC LD40 ; NO... CONTINUE LOAD
00884 07FC ;
00885 07FC 20 0D 0A JSR UNTLK ; CLOSE CHANNEL
00886 07FF 20 8A 08 JSR CLSE1 ; CLOSE THE FILE
00887 0802 90 03 BCC LD180 ; BRANCH ALWAYS
00888 0804 ;
00889 0804 4C C3 08 LD90 JMP ERRDR4 ; FILE NOT FOUND
00890 0807 ;
00891 0807 ;
00892 0807 18 LD180 CLC ; GOOD EXIT
00893 0808 ;
00894 0808 ; SET UP END LOAD ADDRESS
00895 0808 ;
00896 0808 A6 0E LDX EAL
00897 080A A4 0F LDY EAH
00898 080C ;
00899 080C 60 LD190 RTS

00901 080D ; SUBROUTINE TO PRINT TO CONSOLE:
00902 080D ;
00903 080D ; SEARCHING [FOR NAME]
00904 080D ;
00905 080D LUKING
00906 080D A0 0C LDY #MS5-MS1 ; "SEARCHING"
00907 080F 20 62 09 JSR MSG
00908 0812 A5 D1 LDA FNLEN

```

LINE#	LOC	CODE	LINE
-------	-----	------	------

00909	0814	F0 11	BEG LD115
00910	0816	A0 17	LDY #MS6-MS1 ; "FOR"
00911	0818	20 62 09	JSR MSG

00913	081B		; SUBROUTINE TO OUTPUT FILE NAME
00914	081B		;
00915	081B		OUTFN
00916	081B	A0 00	LDY #0
00917	081D	B1 DA	LD110 LDA (FNADR)Y
00918	081F	20 81 09	JSR BSOUT
00919	0822	C8	INY
00920	0823	C4 D1	CPY FNLEN
00921	0825	D0 F6	BNE LD110
00922	0827		;
00923	0827	60	LD115 RTS

00925	0828		; SUBROUTINE TO PRINT:
00926	0828		;
00927	0828		; LOADING/VERIFING
00928	0828		;
00929	0828	A0 39	LOADING LDY #MS10-MS1 ; ASSUME 'LOADING'
00930	082A	4C 62 09	LD410 JMP SPMSG

LINE#	LOC	CODE	LINE
00932	082D		; *****
00933	082D		; * SAVE
00934	082D		; *
00935	082D		; * SAVES TO CASSETTE 1 OR 2, OR
00936	082D		; * IEEE DEVICES 4>=N>=31 AS SELECT-
00937	082D		; * ED BY VARIABLE FA.
00938	082D		; *
00939	082D		; *START OF SAVE IS MEMSTR... END OF
00940	082D		; *SAVE IS .X..Y
00941	082D		; *****
00943	082D	A5 D4	SAVE LDA FA ***MONITOR ENTRY
00944	082F	C9 04	CMR #4
00945	0831	B0 03	BCS SV20
00946	0833		;
00947	0833	4C D2 08	SV10 JMP ERROR9 ;BAD DEVICE #
00948	0836		;
00949	0836		SV20
00950	0836	A9 61	LDA #\$61
00951	0838	B5 D3	STA SA
00952	083A	A4 D1	LDY FNLEN
00953	083C	D0 03	BNE SV25
00954	083E		;
00955	083E	4C CF 08	JMP ERROR8 ;MISSING FILE NAME
00956	0841		;
00957	0841	20 1B 0A	SV25 JSR OPENI
00958	0844	20 A1 08	JSR SAVING
00959	0847	A5 D4	LDA FA
00960	0847	20 9F 09	JSR LISTN
00961	084C	A5 D3	LDA SA
00962	084E	20 AF 09	JSR SECND
00963	0851	A0 00	LDY #0
00964	0853	A5 08	LDA STAH
00965	0855	B5 0D	STA SAH
00966	0857	A5 0A	LDA STAL
00967	0859	B5 0C	STA SAL
00968	085B	20 BF 09	JSR CIOUT
00969	085E	A5 0D	LDA SAH
00970	0860	20 BF 09	JSR CIOUT
00971	0863	38	SV30 SEC
00972	0864	A5 0C	LDA SAL
00973	0866	E5 0E	SBC EAL
00974	0868	A5 0D	LDA SAH
00975	086A	E5 0F	SBC EAH
00976	086C	B0 19	BCS SV50 ;HAVE REACHED END
00977	086E	B1 0C	LDA (SAL)Y
00978	0870	20 BF 09	JSR CIOUT
00979	0873	20 AA 08	JSR STOP
00980	0876	D0 07	BNE SV40
00981	0878		
00982	0878	20 8A 08	BREAK JSR CLSEI
00983	087B	A9 00	LDA #0
00984	087D	38	SEC

LINE#	LOC	CODE	LINE
00985	087E	60	RTS
00986	087F		;
00987	087F	E6 0C	SV40 INC SAL
00988	0881	D0 02	BNE INCR
00989	0883	E6 0D	INC SAH
00990	0885		INCR
00991	0885	D0 DC	BNE SV30
00992	0887	20 CF 09	SV50 JSR UNLSN
00994	088A	24 D3	CLSEI BIT SA
00995	088C	30 11	BMI CLSEI2
00996	088E	A5 D4	LDA FA
00997	0890	20 9F 09	JSR LISTN
00998	0893	A5 D3	LDA SA
00999	0895	29 EF	AND #EF
01000	0897	09 E0	ORA #E0
01001	0899	20 AF 09	JSR SECND
01002	089C	20 CF 09	JSR UNLSN
01003	089F		;
01004	089F	18	CLSEI2 CLC
01005	08A0	60	RTS
01008	08A1		; SUBROUTINE TO OUTPUT:
01009	08A1		; 'SAVING <FILE NAME>'
01010	08A1		;
01011	08A1		SAVING
01012	08A1		;
01013	08A1	A0 41	LDY #MS11-MS1 ; 'SAVING'
01014	08A3	20 62 09	JSR MSG
01015	08A6	20 1B 08	JSR DUTFN ; <FILE NAME>
01016	08A9		;
01017	08A9	60	SAV100 RTS

LINE#	LOC	CODE	LINE
01019	0BAA		; *****
01020	0BAA		; * STOP -- CHECK STOP KEY FLAG AND *
01021	0BAA		; * RETURN Z FLAG SET IF FLAG TRUE. *
01022	0BAA		; * ALSO CLOSES ACTIVE CHANNELS AND *
01023	0BAA		; * FLUSHES KEYBOARD QUEUE. *
01024	0BAA		; * ALSO RETURNS KEY DOWNS FROM LAST *
01025	0BAA		; * KEYBOARD ROW IN .A. ?
01026	0BAA		; *****
01027	0BAA	A5 9B	STOP LDA STKEY ; VALUE OF LAST ROW
01028	0BAC	C9 EF	CMP #SEF ; CHECK STOP KEY POSITION
01029	0BAE	D0 09	BNE STOP2 ; NOT DOWN
01030	0BB0	0B	PHP
01031	0BB1	20 91 09	JSR CLRCH ; CLEAR CHANNELS
01032	0BB4	A9 00	LDA #0
01033	0BB6	85 9E	STA NDX ; FLUSH QUEUE
01034	0BB8	2B	PLP
01035	0BB9	60	STOP2 RTS
01037	0BBA		; *****
01038	0BBA		; * *
01039	0BBA		; * ERROR HANDLER *
01040	0BBA		; * *
01041	0BBA		; * WITH ERROR # IN .A AND CARRY. *
01042	0BBA		; * *
01043	0BBA		; *****
01044	0BBA		;
01045	0BBA	A9 01	ERROR1 LDA #1 ; TOO MANY FILES
01046	0BBC	2C	.BYT #2C
01047	0BBD	A9 02	ERROR2 LDA #2 ; FILE OPEN
01048	0BBF	2C	.BYT #2C
01049	0BC0	A9 03	ERROR3 LDA #3 ; FILE NOT OPEN
01050	0BC2	2C	.BYT #2C
01051	0BC3	A9 04	ERROR4 LDA #4 ; FILE NOT FOUND
01052	0BC5	2C	.BYT #2C
01053	0BC6	A9 05	ERROR5 LDA #5 ; DEVICE NOT PRESENT
01054	0BC8	2C	.BYT #2C
01055	0BC9	A9 06	ERROR6 LDA #6 ; NOT INPUT FILE
01056	0BCB	2C	.BYT #2C
01057	0BCC	A9 07	ERROR7 LDA #7 ; NOT OUTPUT FILE
01058	0BCE	2C	.BYT #2C
01059	0BCE	A9 08	ERROR8 LDA #8 ; MISSING FILE NAME
01060	0BD1	2C	.BYT #2C
01061	0BD2	A9 09	ERROR9 LDA #9 ; BAD DEVICE #
01062	0BD4		;
01063	0BD4	4B	PHA ; ERROR NUMBER ON STACK
01064	0BD5	20 91 09	JSR CLRCH ; RESTORE I/O CHANNELS
01065	0BD8		;
01066	0BD8	A0 00	LDY #MS1-MS1
01067	0BDA		;
01068	0BDA	20 62 09	JSR MSG ; PRINT "CBM I/O ERROR #"
01069	0BDD	6B	PLA

ERROR HANDLER..... PAGE 0026

LINE#	LOC	CODE	LINE
01070	08DE	48	PHA
01071	08DF	09 30	DRA ##30 ; MAKE ERROR # ASCII
01072	08E1	20 B1 09	JSR BSDOUT ; PRINT IT
01073	08E4		;
01074	08E4	68	EREXIT PLA
01075	08E5	38	SEC
01076	08E6	60	RTS

MESSAGES..... PAGE 0027

LINE#	LOC	CODE	LINE
01078	08E7	0D	MS1 .BYT \$D, 'I/O ERROR ', \$A3
01078	08E8	49 2F	
01078	08F2	A3	
01079	08F3	0D	MS5 .BYT \$D, 'SEARCHING ', \$A0
01079	08F4	53 45	
01079	08FD	A0	
01080	08FE	46 4F 52	MS6 .BYT 'FOR ', \$A0
01080	0901	A0	
01081	0902	0D	MS7 .BYT \$D, 'PRESS PLAY ', \$A0
01081	0903	50 52	
01081	090D	A0	
01082	090E	26 20	MS8 .BYT '& RECORD ', \$A0
01082	0916	A0	
01083	0917	4F 4E	MS9 .BYT 'ON TAPE ', \$A3
01083	091F	A3	
01084	0920	0D	MS10 .BYT \$D, 'LOADIN ', \$C7
01084	0921	4C 4F	
01084	0927	C7	
01085	0928	0D	MS11 .BYT \$D, 'SAVING ', \$A0
01085	0929	53 41	
01085	092F	A0	
01086	0930	0D	MS21 .BYT \$D, 'VERIFYIN ', \$C7
01086	0931	56 45	
01086	0939	C7	
01087	093A	0D	MS17 .BYT \$D, 'FOUND ', \$A0
01087	093B	46 4F	
01087	0940	A0	
01088	0941	0D	MS18 .BYT \$D, 'OK ', \$8D
01088	0942	4F 4B	
01088	0944	8D	
01089	0945	0D	MS34 .BYT \$D, '***CBM MONITOR 1.0*** ', \$8D
01089	0946	2A 2A	
01089	095B	8D	
01090	095C	0D	MS36 .BYT \$D, 'BREA ', \$CB
01090	095D	42 52	
01090	0961	CB	
01092	0962		,PRINT MESSAGE TO SCREEN ONLY IF
01093	0962		; OUTPUT ENABLED
01094	0962		;
01095	0962		SPMSG
01096	0962	89 E7 08	MSG LDA MS1,Y
01097	0963	08	PHP
01098	0966	29 7F	AND #\$7F
01099	0968	20 B1 09	JSR BSOUT
01100	096B	CB	INY
01101	096C	28	PLP
01102	096D	10 F3	BPL MSG
01103	096F	18	MS610 CLC
01104	0970	60	RTS

LINE#	LOC	CODE	LINE
01106	0971		;
01107	0971		BASIN
01108	0971	A9 00	LDA #0
01109	0973	8D F0 FF	STA \$FFFF
01110	0976	20 CF FF	JSR \$FFCF
01111	0979	48	PHA
01112	097A	A5 09	LDA MEMMAP
01113	097C	8D F0 FF	STA \$FFFF
01114	097F	68	PLA
01115	0980	60	RTS
01117	0981		BSOUT
01118	0981	48	PHA
01119	0982	A9 00	LDA #0
01120	0984	8D F0 FF	STA \$FFFF
01121	0987	68	PLA
01122	0988	20 D2 FF	JSR \$FFD2
01123	098B	A5 09	LDA MEMMAP
01124	098D	8D F0 FF	STA \$FFFF
01125	0990	60	RTS
01127	0991	A9 00	CLRCH LDA #0
01128	0993	8D F0 FF	STA \$FFFF
01129	0996	20 A6 F2	JSR RCLRCH
01130	0999	A5 09	LDA MEMMAP
01131	099B	8D F0 FF	STA \$FFFF
01132	099E	60	RTS
01134	099F	85 D4	LISTN STA FA
01135	09A1	A9 00	LDA #0
01136	09A3	8D F0 FF	STA \$FFFF
01137	09A6	20 D5 F0	JSR RLISTN
01138	09A9	A5 09	LDA MEMMAP
01139	09AB	8D F0 FF	STA \$FFFF
01140	09AE	60	RTS
01142	09AF	48	SECND PHA
01143	09B0	A9 00	LDA #0
01144	09B2	8D F0 FF	STA \$FFFF

LINE#	LOC	CODE	LINE
01145	09B5	68	PLA
01146	09B6	20 43 F1	JSR RSECND
01147	09B9	A5 09	LDA MEMMAP
01148	09BB	8D F0 FF	STA \$FFFO
01149	09BE	60	RTS

01151	09BF	48	C1OUT PHA
01152	09C0	A9 00	LDA #0
01153	09C2	8D F0 FF	STA \$FFFO
01154	09C5	68	PLA
01155	09C6	20 9E F1	JSR RC1OUT
01156	09C9	A5 09	LDA MEMMAP
01157	09CB	8D F0 FF	STA \$FFFO
01158	09CE	60	RTS

01160	09CF		UNLSN
01161	09CF	A9 00	LDA #0
01162	09D1	8D F0 FF	STA \$FFFO
01163	09D4	20 89 F1	JSR RUNLSN
01164	09D7	A5 09	LDA MEMMAP
01165	09D9	8D F0 FF	STA \$FFFO
01166	09DC	60	RTS

01168	09DD	85 D4	TALK STA FA
01169	09DF	A9 00	LDA #0
01170	09E1	8D F0 FF	STA \$FFFO
01171	09E4	20 D2 F0	JSR RTALK
01172	09E7	A5 09	LDA MEMMAP
01173	09E9	8D F0 FF	STA \$FFFO
01174	09EC	60	RTS

01176	09ED	48	TKSA PHA
01177	09EE	A9 00	LDA #0
01178	09F0	8D F0 FF	STA \$FFFO
01179	09F3	68	PLA
01180	09F4	20 93 F1	JSR RTKSA
01181	09F7	A5 09	LDA MEMMAP
01182	09F9	8D F0 FF	STA \$FFFO
01183	09FC	60	RTS

MESSAGES..... PAGE 0030

LINE# LOC CODE LINE

```

01185 09FD          ACPTR
01186 09FD A9 00    LDA #0
01187 09FF 8D F0 FF STA $FFFF
01188 0A02 20 C0 F1 JSR RACPTR
01189 0A05 4B        PHA
01190 0A06 A5 09    LDA MEMMAP
01191 0A08 8D F0 FF STA $FFFF
01192 0A08 68        PLA
01193 0A0C 60        RTS

```

```

01195 0A0D          UNTLK
01196 0A0D A9 00    LDA #0
01197 0A0F 8D F0 FF STA $FFFF
01198 0A12 20 AE F1 JSR RUNTLK
01199 0A15 A5 09    LDA MEMMAP
01200 0A17 8D F0 FF STA $FFFF
01201 0A1A 60        RTS

```

```

01203 0A1B 4B        PHA
01204 0A1C A9 00    LDA #0
01205 0A1E 8D F0 FF STA $FFFF
01206 0A21 68        PLA
01207 0A22 20 A5 F4 JSR ROPENI
01208 0A25 A5 09    LDA MEMMAP
01209 0A27 8D F0 FF STA $FFFF
01210 0A2A 60        RTS
01211 0A2B          . END

```

ERRORS = 00000

SYMBOL TABLE

SYMBOL	VALUE						
A4	0616	A5	061B	A6	0627	A9	062B
ACC	0003	ACPTR	09FD	ALTM	060F	ALTR	05F5
ALTRIT	0545	ASC1	06F4	ASCII	06ED	B1	0492
B3	049B	B5	04A4	BAD	0100	BASIN	0971
BEGS1	05F1	BREAK	087B	BSOUT	09B1	BUF	0200
CBINV	0092	CINV	0090	CIDUT	09BF	CLRCH	0991
CLSEI	088A	CLSEI2	089F	CMDEND	0531	CMDS	0516
CR	000D	CRLF	0553	D2	0575	DISK	0748
DISK10	0767	DISK15	076A	DISK20	0776	DISK25	0787
DISK5	0764	DM	0599	DM1	059D	DM2	05AF
DSP1	05CC	DSP10	05C9	DSPLYM	05B4	DSPLYR	0573

SYMBOL TABLE

SYMBOL VALUE

EAH	000F	EAL	000E	EREXIT	08E4	ERRL	0638
ERRORP	04AD	ERROR1	08BA	ERROR2	08BD	ERROR3	08C0
ERROR4	08C3	ERROR5	08C6	ERROR6	08C9	ERROR7	08CC
ERROR8	08CF	ERROR9	08D2	ERRS1	05F2	FA	00D4
FLGS	0002	FNADR	00DA	FNLEN	00D1	Q1	063F
G2	064D	G0	0632	HEX09	0744	HEX1T	073A
INCR	08B5	INIT	040F	INWH	0007	INVL	0008
IRQ	0440	L1	0663	L12	06A4	L13	06B3
L14	06CE	L15	06D9	L2	06AE	L20	06C5
L3	0670	L4	0680	L3	06B4	L8	06BD
L9	0694	LD	065E	LD10	07A0	LD110	081D
LD115	0827	LD180	0807	LD190	080C	LD20	07A3
LD25	07AE	LD40	07D5	LD410	082A	LD45	07E3
LD60	07F2	LD64	07F8	LD90	0804	LISTN	099F
LOAD	0796	LOADING	0828	LUKING	080D	MAPPER	062C
MEMMAP	0009	MS1	08E7	MS10	0920	MS11	0928
MS17	093A	MS18	0941	MS21	0930	MS34	0945
MS36	095C	MS5	08F3	MS6	08FE	MS7	0902
MS8	090E	MS9	0917	MS0	0962	MSG10	096F
NDX	009E	NMI	0459	OPENI	0A18	OUTFN	081B
OUTGST	0550	PCH	0000	PCL	0001	PREND	0655
PUTP	0531	RACPTR	F1C0	RCIOUT	F19E	RCLRCH	F2A6
RDOA	0709	RDOA2	0715	RDOB	0716	RDOB4	0739
RDOC	0745	REGK	0558	RLISTN	F0D5	ROPENI	F4A5
RSECND	F143	RTALK	F0D2	RTIP	0472	RTKSA	F193
RUNLSN	F189	RUNTLK	F1AE	S0	04C4	S1	04D2
S2	04E6	S3	04EF	S4	0500	S6	0515
SA	00D3	SAH	000D	SAL	000C	SAV100	08A9
SAVE	082D	SAVING	08A1	SAVX	0015	SECND	09AF
SETIRQ	042B	SETR	053A	SP	0006	SPACE	054D
SPMS0	0962	ST1	048D	STAH	000B	STAL	000A
STATUS	0096	STKEY	009B	STOP	08AA	STOP2	08B9
STR1M1	04B1	SV10	0833	SV20	0836	SV25	0841
SV30	0863	SV40	087F	SV50	0887	SYS	009E
T2T2	06F9	T2T21	06FB	TALK	09DD	TIMB	0488
TIMC	047A	TKSA	09ED	TMPO	0010	TMP2	0012
TPMA	0017	TMPC	0014	TMPPS	0018	UNLSN	09CF
UNTLK	0A0D	WRAP	0016	WROA	06DC	WROB	06E1
XR	0004	YR	0005				

END OF ASSEMBLY

LINE#	LOC	CODE	LINE
-------	-----	------	------

00001	0000		; *****
00002	0000		; *
00003	0000		; * AAA DDDD DDDD OOO N N *
00004	0000		; * A A D D D D O O N N *
00005	0000		; * A A D D D D *** O O N N *
00006	0000		; * A A A A D D D D *** O O N N *
00007	0000		; * A A D D D D O O N N *
00008	0000		; * A A DDDD DDDD OOO N N *
00009	0000		; *
00010	0000		; * L OOO AAA DDDD EEEEE RRRR *
00011	0000		; * L O O A A D D E R R *
00012	0000		; * L O O A A D D E R R *
00013	0000		; * L O O A A A A D D EEEEE RRRR *
00014	0000		; * L O O A A D D E R R *
00015	0000		; * LLLL OOO A A DDDD EEEEE R R *
00016	0000		; *
00017	0000		; *****

00019	0000		; *****
00020	0000		; *
00021	0000		; * 64K-ADD-ON LOADER *
00022	0000		; *
00023	0000		; * THIS PROGRAM INPUTS A FILE NAME FROM *
00024	0000		; * THE KEYBOARD AND ATTEMPTS TO LOAD *
00025	0000		; * THAT PROGRAM INTO ADD-ON RAM SELECTED *
00026	0000		; * BY CONTROL REGISTER VALUE %10000000. *
00027	0000		; * A CBM DISK UNIT #8 IS ASSUMED. AN *
00028	0000		; * AUTO START IS ATTEMPTED THROUGH *
00029	0000		; * (\$FFFC). A NULL FILE NAME TERMINATES *
00030	0000		; * THIS PROGRAM. *
00031	0000		; *
00032	0000		; * 2FEB81 RJF 29JUL81 RJF *
00033	0000		; * CBM PART # 118002 REV B *
00034	0000		; *
00035	0000		; *****

EQUATES.....PAGE 0002

LINE# LOC CODE LINE

00037 0000 * =0
00038 0000 MEMMAP *-**+1 ; COPY OF ADD-ON REQ

00040 0001 ;EQUATES FOR BASIC 4.0
00041 0001 ;
00042 0001 BUF =%200
00043 0001 FNADR =%DA
00044 0001 FNLEN =%D1
00045 0001 STATUS =%96
00046 0001 FA =%D4
00047 0001 SA =%D3
00048 0001 RCLRCH =%F2A6
00049 0001 RLSTN =%F0D5
00050 0001 RSECND =%F143
00051 0001 RCIOUT =%F19E
00052 0001 RUNLSN =%F1B9
00053 0001 RTALK =%F0D2
00054 0001 RTKSA =%F193
00055 0001 RACPTR =%F1C0
00056 0001 RUNTLK =%F1AE
00057 0001 ROPENI =%F4A5

LINE#	LOC	CODE	LINE
00059	0001		;
00060	0001		* =1024
00061	0400		SYS =158 ; BASIC TOKEN VALUE
00062	0400		;
00063	0400		; ENTER COMPILED BASIC TEXT
00064	0400		;
00065	0400	00	. BYT 0
00066	0401	0D 04	. WDR INIT-2
00067	0403	10 00	. WDR \$10
00068	0405	9E	. BYT SYS
00069	0406	2B 31	. BYT '(1039)', 0, 0, 0
00069	040C	00	
00069	040D	00	
00069	040E	00	
00070	040F		;
00071	040F		; SETUP FILE LOAD PARMS
00072	040F		;
00073	040F	A9 80	INIT LDA #X10000000
00074	0411	85 00	STA MEMMAP
00075	0413		;
00076	0413	A0 17	LDY #MS2-MS1
00077	0415	20 90 05	JSR MSG ; TYPE ENTRY MESSAGE
00078	0418		;
00079	0418	A9 00	INIT10 LDA #<BUF
00080	041A	85 DA	STA FNADR
00081	041C	A7 02	LDA #>BUF
00082	041E	85 DB	STA FNADR+1 ; PLACE FOR FILE NAME
00083	0420		;
00084	0420	A9 08	LDA #8
00085	0422	85 D4	STA FA ; DEVICE #8
00086	0424		;
00087	0424		; INPUT FILE NAME FROM SCREEN TO BUF
00088	0424		;
00089	0424	A2 00	LDX #0
00090	0426	20 CF FF	INIT20 JSR \$FFCF
00091	0429	C9 0D	CMP #0D
00092	042B	F0 06	BEG INIT30
00093	042D	9D 00 02	STA BUF, X
00094	0430	E8	INX
00095	0431	D0 F3	BNE INIT20
00096	0433		;
00097	0433	86 D1	INIT30 STX FNLEN
00098	0435	A9 0D	LDA #0D
00099	0437	20 D2 FF	JSR \$FFD2 ; ECHO CR
00100	043A		;
00101	043A	A5 D1	LDA FNLEN
00102	043C	D0 01	BNE INIT40 ; NAME WAS NOT NULL
00103	043E		;
00104	043E	60	RTS
00105	043F		;
00106	043F	78	INIT40 SEI ; PROTECTS ADD-ON
00107	0440		;

LINE#	LOC	CODE	LINE
00109	0440		
00110	0440	A9 00	LOAD LDA #0
00111	0442	85 96	STA STATUS
00112	0444		
00113	0444	A9 60	LDA ##60
00114	0446	85 D3	STA SA ; SPECIAL LOAD COMMAND
00115	0448		
00116	0448	20 3D 05	JSR OPENI ; OPEN THE FILE
00117	0448		
00118	0448	A5 D4	LDA FA
00119	044D	20 FF 04	JSR TALK ; ESTABLISH CHANNEL
00120	0450		
00121	0450	A5 D3	LDA SA
00122	0452	20 0F 05	JSR TKSA ; TELL IT TO LOAD
00123	0455		
00124	0455	20 1F 05	JSR ACPTR ; GET FIRST BYT
00125	0458	8D 78 04	STA EAL ; LOAD ADDRESS LOW
00126	0458		
00127	0458	A5 96	LDA STATUS
00128	045D	4A	LSR A
00129	045E	4A	LSR A
00130	045F	80 41	BCS LD90 ; FILE NOT FOUND
00131	0461		
00132	0461	20 1F 05	JSR ACPTR
00133	0464	8D 79 04	STA EAH ; LOAD ADDRESS HIGH
00134	0467		
00135	0467	A9 FD	LD40 LDA ##FD ; MASK OFF TIMEOUT
00136	0469	25 96	AND STATUS
00137	0468	85 96	STA STATUS
00138	046D		
00139	046D	20 1F 05	LD45 JSR ACPTR ; DATA BYTE
00140	0470	AA	TAX
00141	0471	A5 96	LDA STATUS ; TIMEOUT?
00142	0473	4A	LSR A
00143	0474	4A	LSR A
00144	0475	80 F0	BCS LD40 ; YES... TRY AGAIN
00145	0477		
00146	0477		EAL ==+1
00147	0477		EAH ==+2
00148	0477	8E 00 F0	STX \$F000 ; OPERAND IS SELF MODIFY
00149	047A		
00150	047A	EE 78 04	LD60 INC EAL ; INC STORE ADDRESS
00151	047D	D0 03	BNE LD70
00152	047F	EE 79 04	INC EAH
00153	0482		
00154	0482	24 96	LD70 BIT STATUS ; EOI?
00155	0484	50 E1	BVC LD40 ; NO...
00156	0486		
00157	0486	20 2F 05	JSR UNTLK ; CLOSE CHANNEL
00158	0489		
00159	0489		; CLOSE FILE
00160	0489		
00161	0489	A5 D4	LDA FA
00162	048B	20 C1 04	JSR LISTN
00163	048E		

LINE# LOC CODE LINE

```

00164 048E A5 D3          LDA SA
00165 0490 29 EF          AND #0EF
00166 0492 09 E0          ORA #0E0
00167 0494 20 D1 04      JSR SECND
00168 0497              ;
00169 0497 20 F1 04      JSR UNLSN
00170 049A              ;
00171 049A              ; I/O PEEK, SCREEN PEEK
00172 049A              ; AND WRITE PROTECT BLOCKS
00173 049A              ;
00174 049A A9 E3          LDA #%11100011
00175 049C 8D F0 FF      STA $FFFO
00176 049F              ;
00177 049F 6C FC FF      JMP ($FFFC)
00178 04A2              ;
00179 04A2 20 B3 04      LD90 JSR CLRCH
00180 04A5 A9 00          LDA #%00000000
00181 04A7 8D F0 FF      STA $FFFO      ; MAP BACK TO ROM
00182 04AA              ;
00183 04AA A0 00          LDY #MS1-MS1
00184 04AC 20 90 05      JSR MSG      ; FILE NOT FOUND ERROR
00185 04AF              ;
00186 04AF 5B            CLI
00187 04B0 4C 1B 04      JMP INIT10      ; RETRY

```

LINE#	LOC	CODE	LINE
00189	04B3		
00190	04B3	A9 00	CLRCH LDA #0
00191	04B5	8D F0 FF	STA \$FFFF
00192	04B8	20 A6 F2	JSR RCLRCH
00193	04B8	A5 00	LDA MEMMAP
00194	04BD	8D F0 FF	STA \$FFFF
00195	04C0	60	RTS
00197	04C1	85 D4	LISTN STA FA
00198	04C3	A9 00	LDA #0
00199	04C5	8D F0 FF	STA \$FFFF
00200	04C8	20 D5 F0	JSR RLSTN
00201	04C8	A5 00	LDA MEMMAP
00202	04CD	8D F0 FF	STA \$FFFF
00203	04D0	60	RTS
00205	04D1	48	SECND PHA
00206	04D2	A9 00	LDA #0
00207	04D4	8D F0 FF	STA \$FFFF
00208	04D7	68	PLA
00209	04D8	20 43 F1	JSR RSECND
00210	04D8	A5 00	LDA MEMMAP
00211	04DD	8D F0 FF	STA \$FFFF
00212	04E0	60	RTS
00214	04E1	48	CIDUT PHA
00215	04E2	A9 00	LDA #C
00216	04E4	8D F0 FF	STA \$FFFF
00217	04E7	68	PLA
00218	04E8	20 9E F1	JSR RCIDUT
00219	04E8	A5 00	LDA MEMMAP
00220	04ED	8D F0 FF	STA \$FFFF
00221	04F0	60	RTS
00223	04F1		
00224	04F1	A9 00	UNLSN LDA #0
00225	04F3	8D F0 FF	STA \$FFFF
00226	04F6	20 B9 F1	JSR RUNLSN
00227	04F9	A5 00	LDA MEMMAP

LINE#	LOC	CODE	LINE
00228	04FB	8D F0 FF	STA \$FFFF
00229	04FE	60	RTS
00231	04FF	85 D4	TALK STA FA
00232	0501	A9 00	LDA #0
00233	0503	8D F0 FF	STA \$FFFF
00234	0506	20 D2 F0	JSR RTALK
00235	0509	A5 00	LDA MEMMAP
00236	050B	8D F0 FF	STA \$FFFF
00237	050E	60	RTS
00239	050F	48	TKSA PHA
00240	0510	A9 00	LDA #0
00241	0512	8D F0 FF	STA \$FFFF
00242	0515	68	PLA
00243	0516	20 93 F1	JSR RTKSA
00244	0519	A5 00	LDA MEMMAP
00245	051B	8D F0 FF	STA \$FFFF
00246	051E	60	RTS
00248	051F		ACPTR
00249	051F	A9 00	LDA #0
00250	0521	8D F0 FF	STA \$FFFF
00251	0524	20 C0 F1	JSR RACPTR
00252	0527	48	PHA
00253	0528	A5 00	LDA MEMMAP
00254	052A	8D F0 FF	STA \$FFFF
00255	052D	68	PLA
00256	052E	60	RTS
00258	052F		UNTLK
00259	052F	A9 00	LDA #0
00260	0531	8D F0 FF	STA \$FFFF
00261	0534	20 AE F1	JSR RUNTLK
00262	0537	A5 00	LDA MEMMAP
00263	0539	8D F0 FF	STA \$FFFF
00264	053C	60	RTS

LINE#	LOC	CODE	LINE
00266	053D	48	OPENI PHA
00267	053E	A9 00	LDA #0
00268	0540	8D F0 FF	STA \$FFFF0
00269	0543	68	PLA
00270	0544	20 A5 F4	JSR ROPENI
00271	0547	78	SEI
00272	0548	A5 00	LDA MEMMAP
00273	054A	8D F0 FF	STA \$FFFF0
00274	054D	60	RTS
00276	054E	0D	MS1 .BYT \$D, '?FILE NOT FOUND ERROR', \$8D
00276	054F	3F 46	
00276	0564	8D	
00277	0565	0D	MS2 .BYT \$D, 18
00277	0566	12	
00278	0567	36 34	.BYT '64K ADD-ON-LOADER 2-12-81'
00279	0580	0D	.BYT \$D, \$D, \$D
00279	0581	0D	
00279	0582	0D	
00280	0583	53 59	.BYT 'SYSTEM NAME?', \$A0
00280	058F	A0	
00281	0590	89 90 05	MSG LDA MSG, Y
00282	0593	08	PHP
00283	0594	29 7F	AND #\$7F
00284	0596	20 D2 FF	JSR \$FFD2
00285	0599	C8	INY
00286	059A	28	PLP
00287	059B	10 F3	BPL MSG
00288	059D	60	RTS
00289	059E		.END

ERRORS = 00000

SYMBOL TABLE

SYMBOL		VALUE					
ACPTR	051F	BUF	0200	CIDUT	04E1	CLRCH	04B3
EAL	0479	EAL	0478	FA	00D4	FNADR	00DA
FNLEN	00D1	INIT	040F	INIT10	0418	INIT20	0426
INIT30	0433	INIT40	043F	LD40	0467	LD45	046D
LD60	047A	LD70	0482	LD90	04A2	LISTN	04C1
LOAD	0440	MEMMAP	0000	MS1	054E	MS2	0565
MSG	0590	OPENI	053D	RACPTR	F1C0	RCIDUT	F19E
RCLRCH	F2A6	RLISTN	F0D5	ROPENI	F4A5	RSECND	F143
RTALK	F0D2	RTKSA	F193	RUNLSN	F1B9	RUNTLK	F1AE
SA	00D3	SECND	04D1	STATUS	0096	SYS	009E
TALK	04FF	TKSA	050F	UNLSN	04F1	UNTLK	052F

END OF ASSEMBLY

LINE#	LOC	CODE	LINE
-------	-----	------	------

0001	0000		; *****
0002	0000		; * EEEEE X X PPPP AAA N N DDDD *
0003	0000		; * E X X P P A A NN N D D *
0004	0000		; * E X X P P A A NN N D D *
0005	0000		; * EEEE X PPPP AAAAA N NN D D *
0006	0000		; * E X X P A A NN NN D D *
0007	0000		; * E X X P A A NN NN D D *
0008	0000		; * EEEEE X X P A A N N DDDD *
0009	0000		; *
0010	0000		; * AAA BBBB AAA SSS IIIII CCC *
0011	0000		; * A A B B A A S S I C C *
0012	0000		; * A A B B A A S I C *
0013	0000		; * A A B B A A S I C *
0014	0000		; * AAAAA BBBB AAAAA S I C *
0015	0000		; * A A B B A A S I C *
0016	0000		; * A A B B A A S S I C C *
0017	0000		; * A A BBBB A A SSS IIIII CCC *
0018	0000		; *
0019	0000		; *****
0020	0000		; *
0021	0000		; *
0022	0000		; *****
0023	0000		; *
0024	0000		; * EXPANDED BASIC *
0025	0000		; *
0026	0000		; * PROGRAM TO APPEND TO THE CBM BASIC *
0027	0000		; * VERSION 4.0, 8032, CHRGET ROUTINES *
0028	0000		; * AND ADD COMMANDS TO CACHE PROGRAMS *
0029	0000		; * IN THE 64K ADD-ON MEMORY. *
0030	0000		; *
0031	0000		; * 14JUL81 RJF (BOB) 30JUL81 (JOHN) *
0032	0000		; *
0033	0000		; *****

LINE#	LOC	CODE	LINE
0035	0000		; SYSTEM VARIABLE DEFINITIONS
0036	0000		;
0037	0000		STATUS =%96 ; SYSTEM STATUS VARIABLE
0038	0000		DS =%13 ; DISK STATUS DESCRIPTOR
0039	0000		CHRGOT =%70 ; BASIC ROUTINE TO GET CHARS
0040	0000		CHRGOT =%76 ; GET LAST CHARACTER
0041	0000		GETPTR =%77 ; POINTER TO CHARACTERS
0042	0000		CHDGOT =%D39F ; ROM COPY OF CHRGOT
0043	0000		NEWSTT =%B74A ; NEW STATEMENT EXEX
0044	0000		LNKPRG =%B4B6 ; LINK BASIC LINES
0045	0000		STXTPT =%B622 ; SET START TEXT POINTER
0046	0000		OP94 =%F4A5 ; OPEN A FILE ON IEEE
0047	0000		READY =%B3FF ; RE-ENTER BASIC
0048	0000		FLDAD =%B60B ; RUN AN OVERLAY
0049	0000		RUNC =%B5E9 ; CLEAR VARIABLES
0050	0000		CLRCH =%FFCC ; CLEAR OPEN CHANNEL
0051	0000		FCLOSE =%F2E2 ; CLOSE A FILE
0052	0000		TALK =%F0D2 ; TELL DEVICE TO TALK
0053	0000		ACPTR =%F1C0 ; GET A BYTE
0054	0000		CLSEI =%F72F ; CLOSE A FILE
0055	0000		UNTLK =%F1AE ; UNTALK
0056	0000		TKSA =%F193 ; SEND SECONDARY ADDR
0057	0000		WSW =%B3
0058	0000		T1 =%B4
0059	0000		T2 =%B5
0060	0000		T3 =%B6
0061	0000		FNLEN =%D1 ; FILE NAME LENGTH
0062	0000		LA =%D2 ; LOGICAL FILE NUMBER
0063	0000		SA =%D3 ; SECONDARY ADDRESS
0064	0000		FA =%D4 ; PRIMARY ADDRESS
0065	0000		FNADDR =%DA ; FILE NAME ADDRESS
0066	0000		INDEX1 =%1F ; TEMP POINTER
0067	0000		INDEX2 =%21 ; TEMP POINTER
0068	0000		EAL =%C9 ; END ADDRESS POINTER
0069	0000		VARTAB =%2A ; VARIABLE TABLE POINTER
0070	0000		MEMSIZ =%34 ; TOP OF MEM POINTER
0071	0000		;
0072	0000		; PROGRAM VARIABLES AND CONSTANTS
0073	0000		;
0074	0000		LOC =%7800
0075	0000		%=LOC
0076	7800	4C 2E 79	JMP G000 ; START PROGRAM
0077	7803		;
0078	7803		;
0079	7803		TABHI
0080	7803	79	.BYT >ZZ1
0081	7804	79	.BYT >ZZ3
0082	7805	7A	.BYT >ZZ4
0083	7806	7A	.BYT >ZZ5
0084	7807	79	.BYT >ZZB
0085	7808		;
0086	7808		TABLO
0087	7808	91	.BYT <ZZ1
0088	7809	FD	.BYT <ZZ3
0089	780A	11	.BYT <ZZ4

LINE#	LOC	CODE	LINE
0090	780B	22	. BYT <ZZ5
0091	780C	7C	. BYT <ZZB
0092	780D		;
0093	780D		NCMD =5
0094	780D	52 4C	CMD . BYT 'RLOEG'
0095	7812		FILE **++22
0096	782B	00	. BYT \$00
0097	7829		ENTRY =25
0098	7829		ENTMAX =225 ; ENTRY#9
0099	7829		FITAB **++256
0100	7929	B1 B7	WHEREF . WOR \$B7B2-1 ; GONE3-3-1
0101	792B		MASTER =\$BB ; AVAILABLE SPACE (THREE BYTES
0102	792B	4C 51 79	JUMP JMP START

LINE#	LOC	CODE	LINE
0104	792E		;
0105	792E		; INIT THE MEMORY MANAGER (SYS TO HERE)
0106	792E		;
0107	792E		GOOD
0108	792E	A2 02	LDX ##02 ; MOVE THREE BYTES
0109	7930	8D 28 79	WEDGE LDA JUMP, X
0110	7933	95 79	STA CHRQOT+3, X
0111	7935	CA	DEX
0112	7936	10 F8	BPL WEDGE
0113	7938		;
0114	7938		TABSET
0115	7938	A2 00	LDX #0
0116	793A	8A	TXA
0117	793B	9D 29 78	TABLP STA FITAB, X
0118	793E	EB	INX
0119	793F	D0 FA	BNE TABLP
0120	7941		;
0121	7941		; SETUP THE START ADDRESS
0122	7941		;
0123	7941	A9 80	LDA ##80
0124	7943	85 8C	STA MASTER+1 ; HIGH BYTE
0125	7945	0A	ASL A
0126	7946	85 8B	STA MASTER ; LOW BYTE
0127	7948	85 8D	STA MASTER+2 ; BANK SELECT
0128	794A	85 34	STA MEMSIZ ; SET MEMSIZ TO LOC
0129	794C	A9 78	LDA #>LOC
0130	794E	85 35	STA MEMSIZ+1
0131	7950	60	RTS

LINE#	LOC	CODE	LINE
0133	7951		;
0134	7951		; THIS IS WHERE WE COME TO DO THE WORK
0135	7951		;
0136	7951		START
0137	7951	C9 21	CMP #'! ; LOOK FOR THE ESCAPE CHARACTER
0138	7953	D0 21	BNE NOTME
0139	7955	20 C9 7A	JSR SAVREG ; SAVE THE REGISTERS
0140	7958		;
0141	7958		; TEST HERE FOR QUOTE MODE ETC.....
0142	7958		;
0143	7958	20 70 00	JSR CHRGET ; GET THE NEXT CHARACTER
0144	7958	48	PHA
0145	795C	F0 14	BEG NOTCMD
0146	795E	A2 04	LDX #NCMD-1
0147	7960		FINDC ; FIND THE COMMAND
0148	7960	DD 0D 7B	CMP CMD, X
0149	7963	D0 0A	BNE AGAIN
0150	7965		;
0151	7965		; PUT COMMAND ADDRESS ON STACK
0152	7965		;
0153	7965	68	PLA ; FIX THE STACK
0154	7966		;
0155	7966	BD 03 7B	LDA TABHI, X ; FOUND COMMAND... GO THERE
0156	7969	48	PHA
0157	796A	BD 0B 7B	LDA TABLO, X
0158	796D	48	PHA
0159	796E	60	STRTS RTS
0160	796F		;
0161	796F		AGAIN
0162	796F	CA	DEX
0163	7970	10 EE	BPL FINDC
0164	7972		;
0165	7972		NOTCMD
0166	7972	20 D0 7A	JSR RESREG
0167	7975	68	PLA ; FIX THE STACK
0168	7976		NOTME
0169	7976	C9 3A	CMP #' :
0170	7978	B0 F4	BCS STRTS
0171	797A	4C 7D 00	JMP CHRGET+7

LINE#	LOC	CODE	LINE
0173	797D		;
0174	797D		; GUIT COMMAND
0175	797D		;
0176	797D		; THIS COMMAND EXITS THE MEMORY MANAGER
0177	797D		; AND RESTORES THE CHRGET ROUTINE.
0178	797D		;
0179	797D		GUIT
0180	797D	20 70 00	JSR CHRGET ; CLEAN UP THE LINE
0181	7980	D0 FB	BNE GUIT
0182	7982	A2 02	LDX #02
0183	7984		REST
0184	7984	8D A2 D3	LDA CHRGDT+3, X
0185	7987	95 79	STA CHRGDT+3, X
0186	7989	CA	DEX
0187	798A	10 FB	BPL REST
0188	798C	20 D0 7A	JSR RESREQ
0189	798F	4C 76 00	JMP CHRGDT

LINE#	LOC	CODE	LINE
0191	7992		;
0192	7992		; LOAD FILE FROM DISK INTO EXPANSION MEMORY
0193	7992		;
0194	7992		RECAL
0195	7992	20 D7 7A	JSR GETFN ; GET THE FILENAME
0196	7995	A5 96	LDA STATUS
0197	7997	D0 53	BNE RECOUT ; BAD STATUS
0198	7999	20 10 7B	JSR GTYPE ; GET THE FILE TYPE
0199	799C	A5 96	LDA STATUS
0200	799E	D0 4C	BNE RECOUT
0201	79A0	A9 52	LDA #'R ; SET UP FOR FILE READ
0202	79A2	9D 12 7B	STA FILE, X
0204	79A5		; OPEN THE FILE
0206	79A5	A9 0E	LDA #14 ; SET THE LOGICAL FILE NUMBER
0207	79A7	85 D2	STA LA
0208	79A9	09 60	ORA #60 ; SET THE SECONDARY ADDRESS
0209	79AB	85 D3	STA SA
0210	79AD	20 A5 F4	JSR OP94 ; OPEN THE FILE
0211	79B0	20 3B 7B	JSR ENTER ; ENTER FILENAME INTO TABLE
0212	79B3	80 3F	BCS RECERR ; BAD...
0213	79B5		;
0214	79B5	A5 D4	LDA FA
0215	79B7	20 D2 F0	JSR TALK
0216	79BA	A5 D3	LDA SA
0217	79BC	20 93 F1	JSR TKSA
0218	79BF		REC10
0219	79BF	20 C0 F1	JSR ACPTR ; GET A CHARACTER
0220	79C2	7B	SEI ; KILL THE IRQ
0221	79C3	20 AE 7B	JSR SELBNK ; SET UP EXP. RAM
0222	79C6	A0 00	LDY #0
0223	79C8	91 8B	STA (MASTER), Y
0224	79CA	A9 00	LDA #00 ; RESET THE BANKS
0225	79CC	8D F0 FF	STA \$FFFO
0226	79CF	5B	CLI ; RESTORE THE IRQ
0227	79D0	20 C6 7B	JSR INCMAS
0228	79D3	A5 96	LDA STATUS ; TERMINATE ON STATUS <> 0
0229	79D5	D0 04	BNE REC20
0230	79D7	90 E6	BCC REC10 ; GO DO MORE BYTES
0231	79D9	80 19	BCS RECERR ; OUT OF MEMORY
0232	79DB	C9 40	CMP #64 ; EOI IS OK
0233	79DD	D0 15	BNE RECERR ; NOT EOI...
0234	79DF	20 AE F1	JSR UNTLK
0235	79E2	20 2F F7	JSR CLSEI ; CLOSE THE FILE
0236	79E5	A0 15	LDY #21
0237	79E7	20 6B 7B	JSR ENTLST ; PUT IN THE END ADDRESS
0238	79EA	A9 00	LDA #0
0239	79EC		RECOUT
0240	79EC	85 96	STA STATUS
0241	79EE	20 D0 7A	JSR RESREG ; RESTORE REGISTERS
0242	79F1	4C 70 00	JMP CHRGET ; BACK TO BASIC
0243	79F4		;
0244	79F4		RECERR
0245	79F4	20 AE F1	JSR UNTLK ; CLOSE THE CHANNEL

EXP-BASIC.SRC..... PAGE 0008

LINE# LOC CODE LINE

0246	79F7	20 2F F7	JBR CLSEI	
0247	79FA	A9 20	LDA #32	; SEND AN ERROR BACK
0248	79FC	D0 EE	BNE REOUT	

LINE#	LOC	CODE	LINE
0250	79FE		LOAD
0251	79FE		;
0252	79FE		; THIS ROUTINE LOADS A FILE INTO THE EXPANSION MEMOR
0253	79FE		;
0254	79FE	20 3A 7A	JSR LOADER ; DO THE LOAD STUFF
0255	7A01	A6 20	LDX INDEX1+1
0256	7A03	B6 2B	STX VARTAB+1
0257	7A05	A6 1F	LDX INDEX1
0258	7A07	B6 2A	STX VARTAB
0259	7A09	20 E9 B5	JSR RUNC ; CLEAR THE VARIABLES
0260	7A0C	20 B6 B4	JSR LNKPRG ; LINK BASIC LINES
0261	7A0F	4C FF B3	JMP READY ; SAY READY
0262	7A12		;

LINE#	LOC	CODE	LINE
-------	-----	------	------

0264	7A12		OVRLY
------	------	--	-------

0265	7A12		;
------	------	--	---

0266	7A12		; OVERLAY AN EXISTING PROGRAM (I.E. LOAD FROM BASIC)
------	------	--	--

0267	7A12		;
------	------	--	---

0268	7A12	20 3A 7A	JSR LOADER ; LOAD THE FILE
------	------	----------	----------------------------

0269	7A15	20 22 B6	JSR STXTPT ; PICK UP THE FIRST LINE
------	------	----------	-------------------------------------

0270	7A18	AD 2A 79	LDA WHERET+1 ; PUT CORRECT ADDRESS ON STACK
------	------	----------	---

0271	7A18	48	PHA
------	------	----	-----

0272	7A1C	AD 29 79	LDA WHERET
------	------	----------	------------

0273	7A1F	48	PHA
------	------	----	-----

0274	7A20	4C 0B B6	JMP FLOAD ; GO RUN IT
------	------	----------	-----------------------

0275	7A23		;
------	------	--	---

LINE#	LOC	CODE	LINE
-------	-----	------	------

0277	7A23		EXEC
0278	7A23		;
0279	7A23		; LOAD AND RUN WITH A CLEAR SO VARIABLES ARE GONE
0280	7A23		;
0281	7A23	20 3A 7A	JSR LOADER ; LOAD THE FILE
0282	7A26	A6 20	LDX INDEX1+1
0283	7A28	86 2B	STX VARTAB+1
0284	7A2A	A6 1F	LDX INDEX1
0285	7A2C	86 2A	STX VARTAB
0286	7A2E	20 E9 B5	JSR RUNC ; CLEAR VARIABLES
0287	7A31	20 B6 B4	JSR LNKPRQ ; LINK LINES
0288	7A34	20 22 B6	JSR STXTPT ; SET POINTER TO FIRST LINE
0289	7A37	4C 4A B7	JMP NEWSTT ; EXEC IT
0290	7A3A		;

LINE#	LOC	CODE	LINE
0292	7A3A		;
0293	7A3A		; ROUTINE TO LOAD FILES INTO LOW RAM FROM 64K
0294	7A3A		;
0295	7A3A		LOADER
0296	7A3A	20 D7 7A	JSR GETFN ; READ THE FILENAME
0297	7A3D	A5 96	LDA STATUS
0298	7A3F	D0 5C	BNE LDERR ; BAD FILENAME....
0299	7A41	20 7D 7B	JSR LUKUP ; LOOKUP THE FILE IN DIRECTORY
0300	7A44	80 57	BCS LDERR ; NOT FOUND...
0301	7A46		;
0302	7A46	98	TYA ; INDEX INTO THE DIR FOR ADDRESS
0303	7A47	18	CLC
0304	7A48	69 12	ADC #18
0305	7A4A	A8	TAY
0306	7A4B	B9 29 7B	LDA FITAB,Y ; START LO
0307	7A4E	85 21	STA INDEX2
0308	7A50	C8	INY
0309	7A51	B9 29 7B	LDA FITAB,Y ; START HI
0310	7A54	85 22	STA INDEX2+1
0311	7A56		;
0312	7A56	C8	INY
0313	7A57	B9 29 7B	LDA FITAB,Y ; GET THE MAPPER CONTROL BYTE
0314	7A5A	85 D1	STA FNLEN
0315	7A5C		;
0316	7A5C	C8	INY
0317	7A5D	B9 29 7B	LDA FITAB,Y ; END LO
0318	7A60	85 C9	STA EAL
0319	7A62	C8	INY
0320	7A63	B9 29 7B	LDA FITAB,Y ; END HI
0321	7A66	85 CA	STA EAL+1
0322	7A68		;
0323	7A68		; LOOP TO LOAD DATA INTO LO RAM
0324	7A68		;
0325	7A68		LODLP
0326	7A68	7B	SEI ; NO IRQ ALLOWED
0327	7A69		LODLP0
0328	7A69	A0 00	LDY ##0
0329	7A6B	A6 D1	LDX FNLEN ; SET UP THE BANKS
0330	7A6D	D0 04	BNE LODLP1
0331	7A6F	A2 80	LDX ##80 ; SELECT 0, 2
0332	7A71	D0 02	BNE STLP
0333	7A73		LODLP1
0334	7A73	A2 8B	LDX ##8B ; SELECT 1, 3
0335	7A75		STLP
0336	7A75	8E F0 FF	STX \$FFFO
0337	7A78		;
0338	7A78		;
0339	7A78	B1 21	LDA (INDEX2),Y ; LOAD ADDRESS LO
0340	7A7A	85 1F	STA INDEX1
0341	7A7C	20 9F 7A	JSR BUMPX
0342	7A7F		;
0343	7A7F	B1 21	LDA (INDEX2),Y ; LOAD ADDRESS HI
0344	7A81	85 20	STA INDEX1+1
0345	7A83	20 9F 7A	JSR BUMPX
0346	7A86		;

LINE#	LOC	CODE	LINE
0347	7A86		LODLP2
0348	7A86	B1 21	LDA (INDEX2),Y ; TRANSFER A BYTE
0349	7A88	91 1F	STA (INDEX1),Y
0350	7A8A		;
0351	7A8A	E6 1F	INC INDEX1 ; BUMP THE DESTINATION POINTER
0352	7ABC	D0 02	BNE LODLP3
0353	7ABE	E6 20	INC INDEX1+1
0354	7A90		LODLP3
0355	7A90	20 9F 7A	JSR BUMPX ; BUMP SOURCE POINTER
0356	7A93	90 F1	BCC LODLP2 ; DONE YET?
0357	7A95		;
0358	7A95	A2 00	LDX #0 ; RESTORE THE ROM'S
0359	7A97	8E F0 FF	STX \$FFFO
0360	7A9A	58	CLI ; WE ARE DONE LET IRQ GO
0361	7A9B	18	CLC
0362	7A9C	60	RTS
0363	7A9D		LDERR ; LOAD ERROR
0364	7A9D	38	SEC
0365	7A9E	60	RTS
0366	7A9F		;
0367	7A9F		; SUBROUTINE TO HANDLE DATA POINTERS TO EXPANSION MEM
0368	7A9F		;
0369	7A9F		BUMPX
0370	7A9F	E6 21	INC INDEX2
0371	7AA1	D0 0F	BNE TSTXX
0372	7AA3	E6 22	INC INDEX2+1
0373	7AA3	D0 08	BNE TSTXX
0374	7AA7	A2 80	LDX #80 ; CROSSED OVER TO OTHER BLOCK
0375	7AA9	86 D1	STX FNLEN
0376	7AA8	86 22	STX INDEX2+1
0377	7AAD	A9 88	LDA #\$10111000
0378	7AAF	8D F0 FF	STA \$FFFO
0379	7AB2		TSTXX ; CHECK FOR END OF FILE
0380	7AB2	A6 C9	LDX EAL
0381	7AB4	E4 21	CPX INDEX2
0382	7AB6	D0 08	BNE DONEX
0383	7AB8	A6 CA	LDX EAL+1
0384	7ABA	E4 22	CPX INDEX2+1
0385	7ABC	D0 02	BNE DONEX
0386	7ABE	38	SEC ; DONE WITH LOAD
0387	7ABF	60	RTS
0388	7AC0		DONEX
0389	7AC0	18	CLC
0390	7AC1	60	RTS
0391	7AC2		;

LINE#	LOC	CODE	LINE
0393	7AC2		;
0394	7AC2		; INCREMENT POINTER TO TEXT
0395	7AC2		;
0396	7AC2		INCPTR
0397	7AC2	E6 77	INC GETPTR
0398	7AC4	D0 02	BNE INCRTS
0399	7AC6	E6 78	INC GETPTR+1
0400	7AC8		INCRTS
0401	7AC8	60	RTS
0402	7AC9		;
0403	7AC9		; SAVE AND RESTORE ROUTINES FOR .A, .X, .Y
0404	7AC9		;
0405	7AC9		SAVREG
0406	7AC9	B5 B3	STA WSW
0407	7ACB	B6 B4	STX T1
0408	7ACD	B4 B5	STY T2
0409	7ACF	60	RTS
0410	7AD0		;
0411	7AD0		RESREG
0412	7AD0	A5 B3	LDA WSW
0413	7AD2	A6 B4	LDX T1
0414	7AD4	A4 B5	LDY T2
0415	7AD6	60	RTS

LINE#	LOC	CODE	LINE
0417	7AD7		;
0418	7AD7		; GET FILENAME FROM USER COMMAND
0419	7AD7		;
0420	7AD7		GETFN
0421	7AD7	20 70 00	JSR CHRGET
0422	7ADA	F0 57	BEQ BADFI ; NOT THERE ERROR
0423	7ADC	C9 22	CMP #' " ; LOOK FOR OPEN QUOTE
0424	7ADE	D0 F7	BNE GETFN
0425	7AE0		;
0426	7AE0	A9 00	LDA #0
0427	7AE2	85 96	STA STATUS
0428	7AE4	AA	TAX
0429	7AE5	AB	TAY
0430	7AE6	A9 12	LDA #<FILE
0431	7AE8	85 DA	STA FNADDR
0432	7AEA	A9 78	LDA #>FILE
0433	7AEC	85 DB	STA FNADDR+1
0434	7AEE		FLOOP
0435	7AEE	20 C2 7A	JSR INCPTR
0436	7AF1	81 77	LDA (GETPTR),Y
0437	7AF3	C9 22	CMP #' " ; LOOK FOR END QUOTE
0438	7AF5	F0 0A	BEQ ENDFI
0439	7AF7	9D 12 78	STA FILE,X
0440	7AFA	EB	INX
0441	7AFB	E0 12	CPX #18
0442	7AFD	90 EF	BCC FLOOP ; GO GET MORE
0443	7AFF	B0 32	BCS BADFI ; TOO LONG
0444	7B01		ENDFI
0445	7B01	86 D1	STX FNLEN ; SAVE LENGTH (INDEX)
0446	7B03	A9 00	LDA #0 ; PAD FILENAME WITH #00 ON RIGHT
0447	7B05	E0 12	CPX #18
0448	7B07	B0 06	BCS ENDFIA
0449	7B09	9D 12 78	STA FILE,X
0450	7B0C	EB	INX
0451	7B0D	D0 F6	BNE F110
0452	7B0F		ENDFIA
0453	7B0F	60	RTS
0454	7B10		;
0455	7B10		; GET THE FILE TYPE
0456	7B10		;
0457	7B10		GTYPE
0458	7B10		GCOMMA
0459	7B10	20 70 00	JSR CHRGET
0460	7B13	F0 1E	BEQ BADFI ; NOT ENOUGH INFO.....
0461	7B15	C9 2C	CMP #' ,
0462	7B17	D0 F7	BNE GCOMMA
0463	7B19	A6 D1	LDX FNLEN ; GET THE CURRENT FILENAME LENG
0464	7B1B	9D 12 78	STA FILE,X
0465	7B1E	9D 14 78	STA FILE+2,X ; PUT COMMA HERE ALSO
0466	7B21	20 70 00	JSR CHRGET ; GET TYPE BYTE (S,U,P,R)
0467	7B24	F0 0D	BEQ BADFI
0468	7B26	EB	INX
0469	7B27	9D 12 78	STA FILE,X ; NO VALUE CHECK IS DONE!!!
0470	7B2A	EB	INX ; SKIP TO PUT IN THE READ OR WR
0471	7B2B	EB	INX ; BUT DO NOT PUT IN HERE

LINE#	LOC	CODE	LINE
0472	7B2C	86 D1	STX FNLEN
0473	7B2E		;
0474	7B2E		; PUT IN THE DEVICE NUMBER CHECK LATER*****
0475	7B2E		; FOR NOW DEFAULT TO B ONLY
0476	7B2E		;
0477	7B2E	A9 08	LDA #8
0478	7B30	85 D4	STA FA
0479	7B32	60	RTS
0480	7B33		;
0481	7B33		BADFI
0482	7B33	A9 01	LDA ##01
0483	7B35	85 96	STA STATUS
0484	7B37	60	RTS
0485	7B38		;
0486	7B38		; ENTER FILE INTO TABLE
0487	7B38		; PUT IN LOAD ADDRESS AND SET STATUS IN CARRY
0488	7B38		
0489	7B38		ENTER
0490	7B38	20 7D 7B	JSR LUKUP ; LOOK FOR FILE IN TABLE
0491	7B38	90 3E	BCC BADENT ; DUP FOUND...
0492	7B3D		;
0493	7B3D		; LOOK FOR BLANK ENTRY
0494	7B3D		;
0495	7B3D	A0 E1	LDY #ENTMAX
0496	7B3F		EMPTY
0497	7B3F	B9 29 7B	LDA FITAB,Y
0498	7B42	F0 09	BEG EMPFND
0499	7B44	98	TYA
0500	7B45	38	SEC
0501	7B46	E9 19	SBC #ENTRY
0502	7B48	AB	TAY
0503	7B49	D0 F4	BNE EMPTY
0504	7B48	F0 2E	BEG BADENT
0505	7B4D		;
0506	7B4D		EMPFND ; FOUND AN ENTRY
0507	7B4D	98	TYA ; POINT INDEX1 TO ENTRY
0508	7B4E	18	CLC
0509	7B4F	69 29	ADC #<FITAB
0510	7B51	85 1F	STA INDEX1
0511	7B53	A9 78	LDA #>FITAB
0512	7B55	69 00	ADC #0
0513	7B57	85 20	STA INDEX1+1
0514	7B59		;
0515	7B59		; PLACE ENTRY INTO DIR
0516	7B59		;
0517	7B59	A0 00	LDY #0
0518	7B58		ENTMOR
0519	7B58	B9 12 7B	LDA FILE,Y
0520	7B5E	C9 2C	CMP #', ; ONLY PUT IN DRIVE # AND NAME
0521	7B60	F0 07	BEG ENTFTST
0522	7B62	91 1F	STA (INDEX1),Y
0523	7B64	C8	INY
0524	7B65	C0 12	CPY #18
0525	7B67	90 F2	BCC ENTMOR
0526	7B69		;

LINE#	LOC	CODE	LINE
0527	7B69		ENTFST
0528	7B69	A0 12	LDY #18
0529	7B6B		;
0530	7B6B		ENTLST ; ENTER END ADDRESS
0531	7B6B	A5 B8	LDA MASTER ; SAVE THE LOAD ADDRESS
0532	7B6D	91 1F	STA (INDEX1),Y
0533	7B6F	C8	INY
0534	7B70	A5 B8	LDA MASTER+1
0535	7B72	91 1F	STA (INDEX1),Y
0536	7B74	C8	INY ; PUT IN THE BANK CONTROL BYTE
0537	7B75	A5 BE	LDA MASTER+3
0538	7B77	91 1F	STA (INDEX1),Y
0539	7B79	18	CLC
0540	7B7A	60	RTS
0541	7B7B		BADENT ; COME HERE FOR BAD EXIT
0542	7B7B	38	SEC
0543	7B7C	60	RTS
0544	7B7D		;
0545	7B7D		; LOOKUP FILE IN DIRECTORY
0546	7B7D		;
0547	7B7D		LUKUP
0548	7B7D	A0 E1	LDY #ENTMAX
0549	7B7F		LUK10
0550	7B7F	B9 29 78	LDA FITAB,Y
0551	7B82	D0 09	BNE LUKCMP
0552	7B84		LUK20
0553	7B84	98	TYA ; MOVE TO NEXT FILE
0554	7B85	38	SEC
0555	7B86	E9 19	SBC #ENTRY
0556	7B88	A8	TAY
0557	7B89	D0 F4	BNE LUK10
0558	7B8B	F0 1F	BEG NOTFND
0559	7B8D		;
0560	7B8D		LUKCMP
0561	7B8D	84 B6	STY T3 ; SAVE .Y WHILE DOING THE COMP
0562	7B8F	A2 00	LDX #0
0563	7B91		LUK30
0564	7B91	BD 12 78	LDA FILE,X
0565	7B94	C9 2C	CMP #, ; COMMA MEANS DONE
0566	7B96	F0 08	BEG FOUND
0567	7B98	D9 29 78	CMP FITAB,Y
0568	7B9B	D0 0A	BNE TRYMOR
0569	7B9D	C8	INY
0570	7B9E	E8	INX
0571	7B9F	E0 12	CPX #18
0572	7BA1	90 EE	BCC LUK30
0573	7BA3		;
0574	7BA3		FOUND ; I FOUND IT
0575	7BA3	A4 B6	LDY T3 ; .Y POINTS TO FILE
0576	7BA5	18	CLC
0577	7BA6	60	RTS
0578	7BA7		TRYMOR ; I AM TRYING TO FIND IT
0579	7BA7	A4 B6	LDY T3
0580	7BA9	4C B4 78	JMP LUK20
0581	7BAC		NOTFND ; I LOST IT

LINE#	LOC	CODE	LINE
0582	7BAC	38	SEC
0583	7BAD	60	RTS
0584	7BAE		;
0585	7BAE		; THIS ROUTINE TURNS ON THE PROPER BANK
0586	7BAE		;
0587	7BAE		SELBNK
0588	7BAE	A6 BD	LDX MASTER+2
0589	7BB0	D0 06	BNE SEL10
0590	7BB2	A2 80	LDX ##80 ; TURN ON BANKS 0 AND 2
0591	7BB4	8E F0 FF	STX \$FFFF
0592	7BB7	60	RTS
0593	7BB8		SEL10
0594	7BB8	A2 88	LDX ##88 ; TURN ON BANKS 1 AND 3
0595	7BB8	8E F0 FF	STX \$FFFF
0596	7BBD	A6 BD	LDX MASTER+2 ; ONE BYTE HAS THE \$40 FLAG
0597	7BBF	70 04	BVS SEL30
0598	7BC1	A2 80	LDX ##80
0599	7BC3	86 BD	STX MASTER+2
0600	7BC5		SEL30
0601	7BC5	60	RTS
0602	7BC6		;
0603	7BC6		; BUMP UP THE MASTER POINTER AND SET THE \$40 FLAG
0604	7BC6		;
0605	7BC6		INCMAS
0606	7BC6	E6 88	INC MASTER ; BUMP POINTERS
0607	7BC8	D0 10	BNE MASRTS
0608	7BCA	E6 BC	INC MASTER+1
0609	7BCC	D0 0C	BNE MASRTS ; CHECK FOR OVERFLOW OF BANK
0610	7BCE	A6 BD	LDX MASTER+2 ; WE WENT OVER
0611	7BD0	D0 0A	BNE INC10 ; OUT OF MEMORY?
0612	7BD2	A2 40	LDX ##40 ; SET FLAG FOR OVERFLOW
0613	7BD4	86 BD	STX MASTER+2
0614	7BD6	A2 80	LDX ##80 ; RESET POINTER HI
0615	7BD8	86 BC	STX MASTER+1
0616	7BDA		MASRTS
0617	7BDA	18	CLC
0618	7BDB	60	RTS
0619	7BDC		INC10 ; OUT OF MEMORY
0620	7BDC	38	SEC
0621	7BDD	60	RTS
0622	7BDE		;
0623	7BDE		;
0624	7BDE		; THESE MUST BE AFTER THE DEFINITION
0625	7BDE		;
0626	7BDE		ZZ1 =RECALL-1 ; RECALL FROM DISK
0627	7BDE		ZZ3 =LOAD-1 ; LOAD LOW MEMORY FROM EXPANSI
0628	7BDE		ZZ4 =OVRLY-1 ; OVERLAY PROGRAM IN MEMORY
0629	7BDE		ZZ5 =EXEC-1 ; EXECUTE WITHOUT OVERLAY
0630	7BDE		ZZ8 =QUIT-1 ; EXIT PROGRAM.
0631	7BDE		.END

TEST3

```
10 REM THIS IS A TEST
15 PRINT"TEST3":FORI=1TO200:NEXT
17 PRINT"PRINT THE NUMBERS FROM 51 TO 70"
18 PRINT"ALONG WITH THE SQUARE-ROOT
19 FORI=1TO200:NEXT
20 FORI=51TO70
30 PRINT,I,SQR(I)
40 NEXT
50 !:E,"TEST4"
```

TEST4

```
10 REM THIS IS A TEST
15 PRINT"TEST4":FORI=1TO200:NEXT
17 PRINT"YOU HAVE SEEN THE SYSTEM LOAD AND"
18 PRINT"EXECUTE THREE PROGRAMS FROM THE
20 PRINT"EXPANSION MEMORY. IF YOU LOOK AT
30 PRINT"THE FILES WITH THE WORD 'TEST' IN THEM
40 PRINT"YOU WILL SEE HOW THIS IS DONE
45 PRINT"TYPE SPACE TO CONTINUE OR QUIT"
50 GETA$:IFA$=""THEN50
60 IFA$="0"THEN!@:END
65 IFA$=" "THEN!E,"TEST1"
70 IFA$="2"THEN!E,"TEST2"
80 IFA$="3"THEN!E,"TEST3"
```

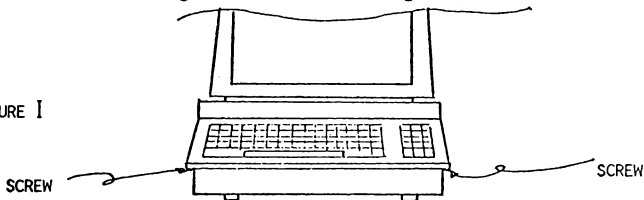
P/N 119001-33A

```
0 ; *****
0 ;# DEMO INSTRUCTIONS #
0 ; *****
0 ;
0 ;MEMORY CHECK PROGRAM
0 ;1.SYSTEM MUST BE AN 8032
0 ;2.INSTALL DISKETTE IN DRIVE 0
0 ;3. TYPE:LOAD"8032.MEM.PRG",08(RETURN)
0 ;    3A. SHIFT RUN/STOP WILL ALSO WORK
0 ;4. IF LOAD USED TYPE:RUN (RETURN)
0 ;5. GRAPHIC DISPLAY SHOULD APPEAR IMMEDIATELY
0 ;    AFTER PROGRAM HAS LOADED AND EXECUTED
0 ;
0 ; ALL DISPLAYS ARE FORMATTED THE SAME :
0 ;000000000: R/W 1 IS READ/WRITE BLOCK ONE,ETC.
0 ;CYCLES IS THE NUMBER OF PASSES FOR THAT 32K.SECTION
0 ;FAILS ARE THE CUMULATIVE FAILS FOR THAT 32K.SECTION
0 ;
0 ; THE SCREEN SHOULD REVERSE APPROXIMATELY EVERY
0 ;8 SECONDS. ON A BRIGHT SCREEN THE LOWER 32K. IS RUNNING
0 ;ON A DARK SCREEN THE UPPER 32K.IS RUNNING
0 ; THE NOTCH IN THE UPPER LEFT CORNER IS FOR SCREEN PEEK
0 ;
0 ; AS THIS PROGRAM EXECUTES ALMOST ENTIRELY FROM THE
0 ;64K.ADD-ON ANY FAILURES IN MEMORY ABOVE F800(HEX)
0 ;WILL CAUSE THE PROGRAM TO CRASH. THE MEMORY CHECK
0 ;IS A 00 TO FF AND AN AA TO 55 TRANSITION CHECK,
0 ;I.E. WRITE 00 CHECK 00,WRITE FF EVERY 4TH.BYTE,CHECK FF
0 ;SWAP FF WITH NEXT LOWEST BYTE,CHECK ETC. THE PROGRAM THEN
0 ;DOES THE SAME THING WITH AA AND 55 FOR A FAIRLY
0 ;COMPREHENSIVE ZERO TO ONE TRANSITION CHECK.
0 ;
0 ;
0 ;DEMO PROGRAM :
0 ;1. TYPE: LOAD"EXPANDED-DEMO",08 (RETURN)
0 ;2. WHEN READY APPEARS,TYPE: RUN (RETURN)
0 ;    PROGRAM EXECUTES THREE PROGRAMS FROM THE ADD-ON
0 ;    WITH COMMENTS
0 ;3. TYPE: LOAD"ADD-ON-LOAD",08 (RETURN)
0 ;4. TYPE: RUN (RETURN)
0 ;5. PET SHOULD RESPOND WITH SYSTEM NAME ?
0 ;6. RESPOND WITH THE TYPE OF BASIC YOU REQUIRE
0 ;    I.E. BASIC4.0/80 (RETURN)
0 ;7. PET SHOULD RESPOND BY LOADING FROM DRIVE ZERO
0 ;    AND GIVING THE STANDARD BOOT DISPLAY. HOWEVER,
0 ;    THE PET IS NOW EXECUTING FROM THE ADD-ON RAM.
0 .END
```

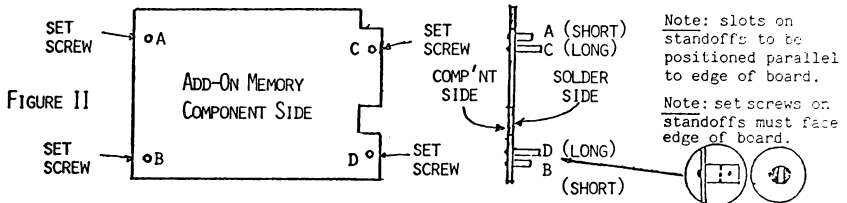
LINE#	LOC	CODE	LINE
0001	0000		;*****
0002	0000		;* TOGGLE PRG. *
0003	0000		;*****
0004	0000		;
0005	0000		;THIS PROGRAM "TOGGLES" MOST OF THE GATES
0006	0000		;WITHIN THE 64K.ADD-ON
0007	0000		;
0008	0000		*=\$0400 ;START MEMORY LOCATION
0009	0400		; ALLOWS "RUN"COMMAND FOR RESTART
0010	0400		;
0011	0400	00	BYTE 0,13,4,10,0,158 ;10 SYS
0011	0401	0D	
0011	0402	04	
0011	0403	0A	
0011	0404	00	
0011	0405	9E	
0012	0406	28 31	BYTE '(1039)',0,0,0
0012	0406	00	
0012	040D	00	
0012	040E	00	
0013	040F	4C 12 04	JMP START
0015	0412		;*****
0016	0412		;* BLOCK 0-2 *
0017	0412		;*****
0018	0412	78	START SEI ;NO INTERRUPTS
0019	0413	A9 80	LDA #\$80 ;ENA 0 & 2
0020	0415	8D F0 FF	STA \$FFFF
0021	0418	8D 00 80	STA \$8000 ;WRITE 0
0022	041B	8D 00 C0	STA \$C000 ;WRITE 2
0023	041E		;*****
0024	041E		;* BLOCK 1-3 *
0025	041E		;*****
0026	041E	A9 8C	LDA #\$8C ;ENA 1& 3
0027	0420	8D F0 FF	STA \$FFFF
0028	0423	8D 00 80	STA \$8000 ;WRITE 1
0029	0426	8D 00 C0	STA \$C000 ;WRITE 3
0030	0429		;*****
0031	0429		;* W.PROT 1-3 *
0032	0429		;*****
0033	0429	A9 8F	LDA #\$8F ;W/P 1 & 3
0034	042B	8D F0 FF	STA \$FFFF
0035	042E	8D 00 80	STA \$8000 ;WRITE 1
0036	0431	8D 00 C0	STA \$C000 ;WRITE 3
0037	0434		;*****
0038	0434		;* W. PRTO 0-2 *
0039	0434		;*****
0040	0434	A9 83	LDA #\$83 ;W/P 0 & 2
0041	0436	8D F0 FF	STA \$FFFF
0042	0439	8D 00 80	STA \$8000 ;WRITE 0
0043	043C	8D 00 C0	STA \$C000 ;WRITE 2
0044	043F		;*****
0045	043F		;* T.V.PEEK 0-2*
0046	043F		;*****
0047	043F	A9 A0	LDA #\$A0 ;ENA PEEK 0

1. Turn Pet off and unplug power outlet.
2. Remove 2 screws holding cover to base. (see figure I).

FIGURE I



3. Assemble 4 standoffs with 4 phillips head screws, and set aside. (see figure II).



4. Using a small chisel head screwdriver, remove device 6502 MPU from position UB14 on the main CPU board. (save 6502)
5. Install flat ribbon cable into UB14 on main CPU board (replacing the removed 6502). Position cable such that it is facing front of computer.
6. Install long power harness to J11 on main CPU board. (note polarity). Position harness towards front of computer.
7. Install short harness to J10 on main CPU board. (note polarity).
8. Ascertain that the 3 cables are positioned towards front of computer.
9. Install the 64K memory add-on board over the main CPU board such that slotted side is on right hand side of computer.
10. Position standoffs such that, the 2 long ones straddle the edge of the computer base, while the 2 short ones straddle the main CPU board heat sink. (see figure III).

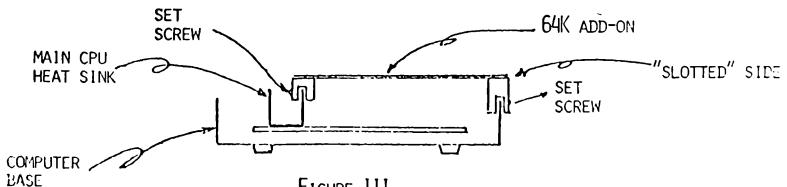
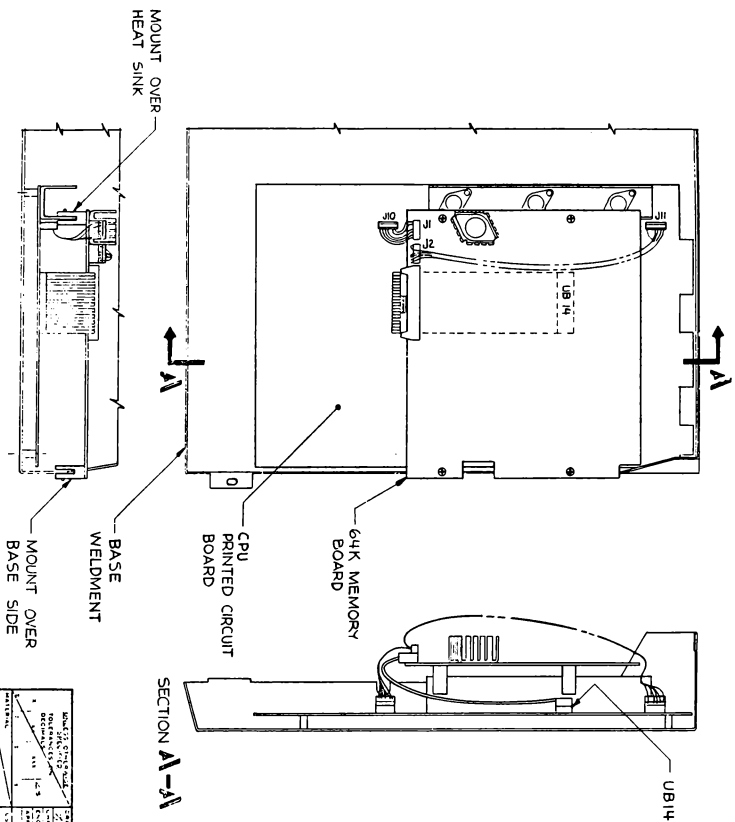


FIGURE III

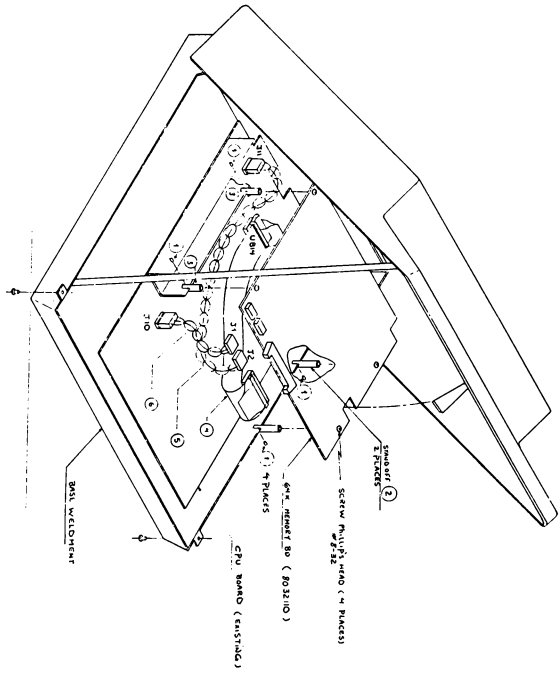
11. With Allen wrench tool tighten 4 sets screws on standoffs to secure add-on board.
12. Connect flat ribbon cable to edge connector on 64K add-on board.
13. Connect the long power harness to J2 on 64K board.(note polarity).
14. Connect the short harness to J1 on the 64K board. (note polarity).
15. Loosen 4 standoff screws and adjust board to it's left most position, retighten 4 screws.
16. Close Pet cover & reinstall 2 screws to secure cover to base.

Note: At times the inside of the cover may rub the edge of the board during opening and closing of the computer cover.

REVISIONS		
LTN	DESCRIPTION	DATE
A	PRODUCTION RELEASE	11/1
B	REV PER ECO 2144	7/6/44

[illegible]

REVISIONS		DATE	BY	APP'D
1	REVISED			
2	REVISED			



NOTES:

- ① PART OF STAIRS
- ② STAIRS (P/M 10/10/01)
- ③ STAIRS (P/M 10/10/01)
- ④ STAIRS (P/M 10/10/01)
- ⑤ STAIRS (P/M 10/10/01)
- ⑥ STAIRS (P/M 10/10/01)
- ⑦ STAIRS (P/M 10/10/01)
- ⑧ STAIRS (P/M 10/10/01)
- ⑨ STAIRS (P/M 10/10/01)
- ⑩ STAIRS (P/M 10/10/01)
- ⑪ STAIRS (P/M 10/10/01)
- ⑫ STAIRS (P/M 10/10/01)
- ⑬ STAIRS (P/M 10/10/01)
- ⑭ STAIRS (P/M 10/10/01)
- ⑮ STAIRS (P/M 10/10/01)
- ⑯ STAIRS (P/M 10/10/01)
- ⑰ STAIRS (P/M 10/10/01)
- ⑱ STAIRS (P/M 10/10/01)
- ⑲ STAIRS (P/M 10/10/01)
- ⑳ STAIRS (P/M 10/10/01)
- ㉑ STAIRS (P/M 10/10/01)
- ㉒ STAIRS (P/M 10/10/01)
- ㉓ STAIRS (P/M 10/10/01)
- ㉔ STAIRS (P/M 10/10/01)
- ㉕ STAIRS (P/M 10/10/01)
- ㉖ STAIRS (P/M 10/10/01)
- ㉗ STAIRS (P/M 10/10/01)
- ㉘ STAIRS (P/M 10/10/01)
- ㉙ STAIRS (P/M 10/10/01)
- ㉚ STAIRS (P/M 10/10/01)
- ㉛ STAIRS (P/M 10/10/01)
- ㉜ STAIRS (P/M 10/10/01)
- ㉝ STAIRS (P/M 10/10/01)
- ㉞ STAIRS (P/M 10/10/01)
- ㉟ STAIRS (P/M 10/10/01)
- ㊱ STAIRS (P/M 10/10/01)
- ㊲ STAIRS (P/M 10/10/01)
- ㊳ STAIRS (P/M 10/10/01)
- ㊴ STAIRS (P/M 10/10/01)
- ㊵ STAIRS (P/M 10/10/01)
- ㊶ STAIRS (P/M 10/10/01)
- ㊷ STAIRS (P/M 10/10/01)
- ㊸ STAIRS (P/M 10/10/01)
- ㊹ STAIRS (P/M 10/10/01)
- ㊺ STAIRS (P/M 10/10/01)
- ㊻ STAIRS (P/M 10/10/01)
- ㊼ STAIRS (P/M 10/10/01)
- ㊽ STAIRS (P/M 10/10/01)
- ㊾ STAIRS (P/M 10/10/01)
- ㊿ STAIRS (P/M 10/10/01)

REVISIONS		DATE	BY	APP'D
1	REVISED			
2	REVISED			